Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

# КВАЛІФІКАЦІЙНА РОБОТА
# Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Мультимодальна система обробки
та аналізу коротких новинних відео
(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-21-2

Захарій Болотніков
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки

(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект

(повна назва освітньої програми)
Керівник доц. Лариса Чала
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Олег ЗОЛОТУХІН
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

# ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Болотнікову Захарію Вячеславовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Мультимодальна система обробки та аналізу коротких новинних відео

затверджена наказом університету від 19 травня 20 25 р. № 378Ст

2. Термін подання студентом роботи до екзаменаційної комісії 18 червня 20 25 р.

3. Вихідні дані до роботи Офіційна документація та посібники з програмування мовою Python і використання її бібліотек (зокрема PyTorch, Transformers, OpenCV), технічна документація фреймворку Flask, специфікації API Google Fact Check, документація систем виявлення маніпуляцій у зображеннях, загальнодоступні анотовані набори даних із твердженнями та відеофейками, архіви результатів перевірок сервісів TinEye та OpenTimestamps, наукові публікації з тематики NLP і дезінформації, а також вебресурси, пов'язані з фактчекінгом, штучним інтелектом і цифровою безпекою.

4. Перелік питань, що потрібно опрацювати в роботі

1) Аналіз предметної галузі та постановка задачі

2) Теоретичні дослідження

3) Проектування системи та її розробка

4) Експериментальні дослідження

**КАЛЕНДАРНИЙ ПЛАН**

| № | Назва етапів роботи | Строк / терміни виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Отримання завдання на кваліфікаційну роботу | 19.05.2025 | виконано |
| 2 | Аналіз предметної галузі та постановка задачі | 20.05.2025 | виконано |
| 3 | Огляд аналогів та опис методів | 21.05.2025 | виконано |
| 4 | Розробка концепції мультимодальної системи | 21.05.2025 | виконано |
| 5 | Розробка, налаштування, реалізація системи | 30.05.2025 | виконано |
| 6 | Створення веб-інтерфейсу системи | 03.06.2025 | виконано |
| 7 | Тестування системи | 04.06.2025 | виконано |
| 8 | Написання пояснювальної записки | 06.06.2025 | виконано |
| 9 | Попередній захист | 07.06.2025 | виконано |
| 10 | Захист перед екзаменаційною комісією | 18.06.2025 | виконано |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

Дата видачі завдання 19 травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Лариса Чала
(підпис) (посада, власне ім'я, прізвище)

# РЕФЕРАТ

Пояснювальна записка: 78 с., 23 рис., 4 табл, 3 дод, 50 джерел.

ВІДЕОНОВИНИ, ДИПФЕЙК, КОМП'ЮТЕРНИЙ ЗІР, МУЛЬТИМОДАЛЬНИЙ АНАЛІЗ, РОЗПІЗНАВАННЯ МОВЛЕННЯ, ФАКТЧЕКІНГ, ФЕЙКОВІ НОВИНИ, LLM.

Об'єкт дослідження – новинні відеоролики тривалістю до п'яти хвилин.

Предмет дослідження – мультимодальна модель машинного навчання, яка здатна виявляти сфабриковані або маніпульовані компоненти у новинних відео шляхом спільного аналізу потоків мовлення, тексту та візуальних даних.

Мета роботи – автоматична ідентифікація фальшивого контенту у новинних відео з використанням найсучасніших методів розпізнавання мовлення, обробки природної мови та комп'ютерного зору.

Методи дослідження – автоматичне розпізнавання мовлення на основі трансформатора; багатомовні моделі для вилучення тверджень та семантичного втягування; класичні та глибоко навчальні алгоритми аналізу зображень (аналіз рівня помилок, кореляція PRNU, легкі CNN, детектори дипфейків); локально-чутливе хешування та пошук приблизного найближчого сусіда для швидкого виявлення дублікатів; статистичне об'єднання рішень.

# ABSTRACT

Bachelor’s thesis contains: 78 pp., 23 fig., 4 tabl., 3 ann., 50 references.

AUTOMATIC SPEECH RECOGNITION, COMPUTER VISION, DEEPFAKE, FACT-CHECKING, FAKE NEWS, LLM, MULTIMODAL ANALYSIS, VIDEO NEWS.

Object of study – news videos up to five minutes long.

Subject of the study – a multimodal machine learning model that detects fabricated or manipulated components in news videos by jointly analyzing speech streams, text, and visual data.

Goal of the work – to automatically identify fake content in news videos using the most advanced methods of speech recognition, natural language processing, and computer vision.

Research methods – transformer-based automatic speech recognition; multilingual language models for statement extraction and semantic embedding; classical and deep learning image analysis algorithms (Error Level Analysis, PRNU correlation, lightweight CNNs, deepfake detectors); locally sensitive hashing and approximate nearest neighbor search for fast duplicate detection; statistical solution pooling.

**ЗМІСТ**

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ANN – Approximate Nearest Neighbor – пошук наближених сусідів;

API – Application Programming Interface – програмний інтерфейс прикладного рівня;

ASR – Automatic Speech Recognition – автоматичне розпізнавання мовлення;

AUC – Area Under the ROC Curve – площа під ROC-кривою;

BM25 – Best Matching 25 – функція оцінювання релевантності BM25;

CNN – Convolutional Neural Network – згорткова нейронна мережа;

CPU – Central Processing Unit – центральний процесор;

DFDC – Deepfake Detection Challenge – набір даних Deepfake Detection Challenge;

Docker – платформа контейнеризації Docker;

ELA – Error Level Analysis – аналіз рівня помилки JPEG;

F1 – F1-Score – гармонічне середнє точності та повноти;

FFmpeg – Fast Forward MPEG – мультимедійний конвертор FFmpeg;

GAN – Generative Adversarial Network – змагальна генеративна мережа;

GPU – Graphics Processing Unit – графічний процесор;

HTTP – HyperText Transfer Protocol – протокол передавання гіпертексту;

HNSW – Hierarchical Navigable Small World – граф HNSW для пошуку сусідів;

JSON – JavaScript Object Notation – текстовий формат обміну даними JSON;

JWT – JSON Web Token – веб-токен JSON;

K8s – Kubernetes – система оркестрації контейнерів Kubernetes;

LUFS – Loudness Units relative to Full Scale – одиниці гучності LUFS;

LLM – Large Language Model – велика мовна модель;

LSH – Locality-Sensitive Hashing – гешування з урахуванням близькості;

MAE – Masked AutoEncoder – автоенкодер з маскуванням;

MiB – Mebibyte – мебібайт ($2^{20}$ байт);

MLP – Multi-Layer Perceptron – багатошаровий перцептрон;

mT5 – multilingual Text-to-Text Transfer Transformer – багатомовна модель mT5;

NLP – Natural Language Processing – обробка природної мови;

PRNU – Photo-Response Non-Uniformity – неуніформність фоточутливості матриці;

ROC – Receiver Operating Characteristic – характеристика роботи приймача;

RTT – Round-Trip Time – час кругового проходження;

SHA-1 – Secure Hash Algorithm 1 – алгоритм гешування SHA-1;

SimHash – Similarity Hash – геш схожості SimHash;

TLS – Transport Layer Security – протокол безпеки TLS;

URI – Uniform Resource Identifier – уніфікований ідентифікатор ресурсу;

URL – Uniform Resource Locator – уніфікований локатор ресурсу;

WER – Word Error Rate – відносна помилка розпізнавання слів;

WebSocket – протокол двосторонніх веб-з’єднань WebSocket.

# ВСТУП

Дешевий відеомонтаж, заміна облич за допомогою ШІ та репости в соціальних мережах у режимі реального часу знизили кардинально ціну створення переконливих новинних підробок. Вірусні кліпи, що показують «ракетні удари» чи «політичні зізнання», можуть накопичити мільйони переглядів задовго до того, як відреагують професійні перевіряльники фактів, посилюючи паніку в населенні, поляризацію та, у випадку України, інформаційну війну. Тому центральна проблема, що розглядається у кваліфікаційній роботі, полягає у швидкому надійному виявленні дезінформації, яка прихована в коротких новинних відео, поданих різними мовами.

Для вирішення цієї задачі робота поділена на чотири етапи. У першому розділі окреслюється рівень загроз, досліджуються типові стилі маніпуляцій (архівне неправильне контекстування, сплайсинг, дипфейки тощо) та визначено чотири потоки даних – аудіо, текст, візуальні матеріали, метадані – які містять потенційні криміналістичні підказки. Також сформульовано мету дослідження: розробка мультимодальної системи перевірки, здатної виносити вердикт щодо правдивості новини протягом відносно короткого часу.

У другому розділі розглянуто та проаналізовано конкретні алгоритми, які можуть вирішити багатокрокову задачу виявлення фейкової інформації. У ньому пояснюється автоматичне розпізнавання мовлення на основі трансформатора, багатомовні кодери типу BERT для класифікації тверджень, моделі сімейства T5 для текстового впровадження, класичні детектори обробки сигналів, такі як аналіз рівня помилок, легкі CNN для виявлення копіювання та переміщення, розпізнавачі дипфейків, локально-чутливе хешування для майже дублікатних кадрів та спільні вбудовування в стилі CLIP, які поєднують вміст зображення з оповіддю.

Розділ 3 перетворює цей методологічний арсенал на повноцінну систему. Мікросервісна архітектура демультиплексує завантажений кліп, запускає транскрипцію Whisper, переклад Marian та перевірку заяв MiniLM у мовній гілці, тоді як двоступеневий каскад бачення – tiny-CNN gatekeeper, Xception deep-fake detector, тести ELA/PRNU, пошук дублікатів – обробляє ключові кадри паралельно. TinEye, Google Images та OpenTimestamps забезпечують зовнішнє походження; баєсівський шар об’єднання інтегрує оцінки модальності та обслуговує багатомовну веб-панель. Вибір дизайну обґрунтовується за допомогою списків коду, діаграм контейнерів та повного робочого процесу розгортання.

В останньому розділі представлені кількісні результати та план дій на майбутнє. Дослідження кількісно визначають внесок кожного модуля, тоді як калібрований бал ілюструє поясненьність. У дорожній карті окреслено монетизацію, автоматичне масштабування Kubernetes, майбутній детектор аудіофейків, підтримку відео з інших соціальних мереж (Meta, X/Twitter, TikTok), реєстр довіри до публічних осіб та push-сповіщення електронною поштою та Telegram. Ці кроки гарантують, що система залишиться як науково актуальною, так і економічно самодостатньою.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

## 1.1 Аналіз предметної галузі

У сьогоднішньому інформаційному середовищі новинні повідомлення, що поширюються через телебачення, радіо, інтернет-ЗМІ, соціальні мережі та месенджери, відіграють ключову роль. Обсяг новинного контенту, особливо відео, зростає експоненційно завдяки стрімкому розвитку цифрових технологій [1]. Саме відеоновини мають високий рівень впливу, оскільки люди часто сприймають візуальну інформацію як більш достовірну, однак з такою довірою до відео пов’язаний і негативний наслідок – це велика загроза поширення не тільки дезінформації та фейкових матеріалів, а зокрема й деструктивних повідомлень (рисунок 1.1).

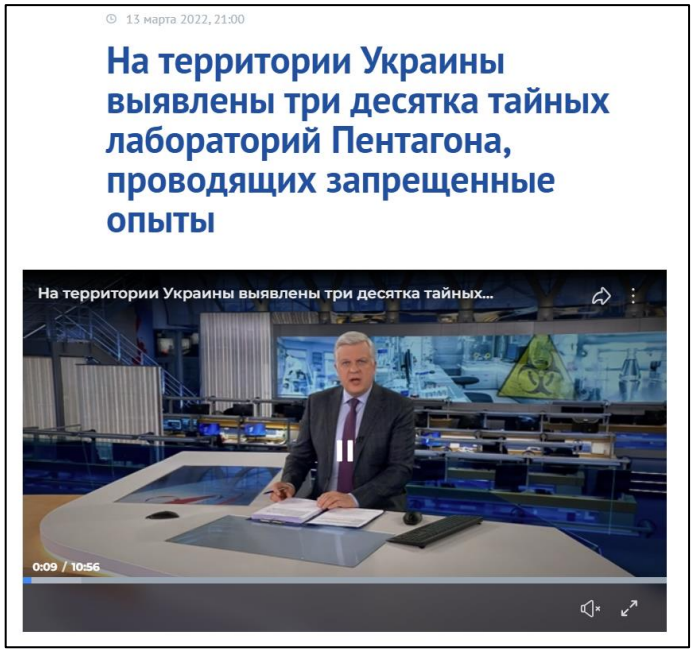


Рисунок 1.1 – Приклад фейкової відеоновини

Для подолання цих викликів потрібен автоматизований аналіз відеоновин задля виявлення фейків та інших форм маніпуляцій. Під фейками мають на увазі викривлення фактів, використання старих архівних

зображень замість поточної події, вставлення змонтованих фрагментів у відеоряд тощо [2]. В останні роки додатково з'явилися різні технології гіперреалістичних підробок, зокрема дипфейків, що дають змогу змінювати обличчя та мову людей у відео за допомогою методів штучного інтелекту (ШІ). При такій ситуації значно ускладняються перевірка достовірності контенту неозброєним оком. Швидкість зросту їх кількості експоненціальна (рисунок 1.2).

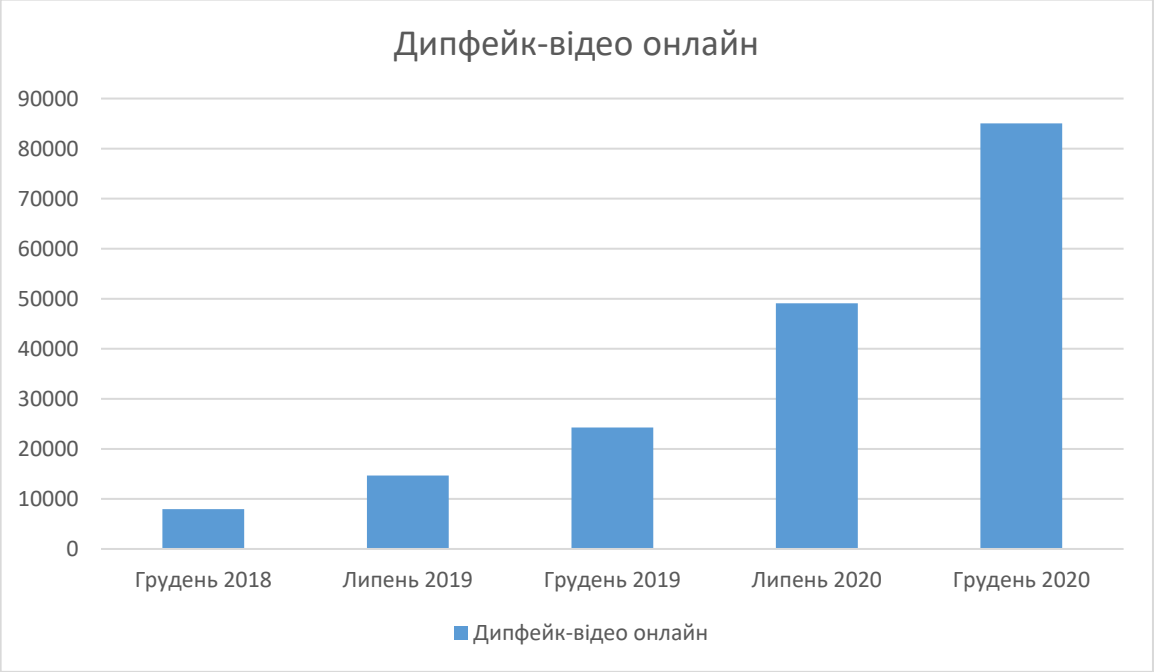


Рисунок 1.2 – Статистика кількості ідентифікованих дипфейк-відео від Sensity AI

Мультимодальний аналіз передбачає обробку декількох типів даних одночасно: аудіо, тексту, відеоряду, зображень, інфографіки, а також метаданих [3]. Також варто зазначити, що з технічної точки зору важливо, що відео – це послідовність зображень (кадрів або фреймів), часто зі звуком і додатковими елементами такими, як субтитри, логотипи тощо. Мультимодальний підхід саме вимагає обробку усіх цих потоків. Тобто ми вже говоримо про паралельний аналіз.

Відповідно, охоплюються наступні методи:

– автоматичне розпізнавання мовлення (ASR), а саме конвертація аудіодоріжки у текст, щоб далі застосовувати методи аналізу природної мови (NLP);

– обробка природної мови, а саме виявлення ознак фейковості за лексико-семантичними, стилістичними та фактологічними показниками [4];

– аналіз зображень та відеокадрів (Computer Vision, CV) для виявлення фотомонтажу, дипфейків, штучних вставок, ШІ-генерованих матеріалів, перевірка EXIF-даних, пошук «випадкових вставок»;

– системи підтвердження справжності – блокчейн, цифрові підписи, пошук оригіналу в інтернет-архівах (TinEye, Google Images).

Саме через поєднання цих підходів і будується комплексна система, здатна оперативно перевіряти достовірність новинного відео. Слід заначити, що ця предметна галузь охоплює багато важливих напрямів. По-перше, журналістику, де використовуються алгоритми ШІ для перевірки матеріалів перед публікацією [5]. По-друге, в аналітиці соціальних мереж, де новини поширюються швидко й часто неконтрольовано, тому потрібно виявляти фейкові відео раніше, ніж вони стануть вірусними. По-третє, у сфері кібербезпеки, бо дипфейки можуть використовуватися для шантажу, провокацій, політичних маніпуляцій тощо. По-четверте, у судовій експертизі, де здійснюється аналіз цифрових доказів у межах розслідувань.

Також є актуальним використання глибоких нейронних мереж. Для розпізнавання мовлення застосовуються моделі, які підтримують різні мови, типу Whisper, DeepSpeech, Vosk, що можуть навіть працювати не тільки онлайн, а і офлайн [1]. У NLP-сфері популярні великі мовні модель (LLM), наприклад, GPT, Gemini, Claude, LLaMA, які аналізують текст га логіку, узгодженість фактів. Для комп’ютерного зору використовують конволюційні нейронні мережі чи трансформери (ViT) для виявлення ознак дипфейкую. Крім того, для порівняння з «оригіналом» застосовують

детектори копипасти, аналіз рівня помилки JPEG (ELA), шумових патернів PRNU, виявдення невідповідностей у метаданих [3], [6].

## 1.2 Обґрунтування актуальності

Актуальність проблеми виявлення фейкових відеоновин пояснюється низкою соціальних і технологічних чинників [2]. По-перше, інформаційні війни стали частиною сучасних конфліктів. Пропагандистські канали чи ворожі інформаційні ресурси поширюють фейки для дестабілізації громадської думки, викривлення подій, дискредитації уряду чи певних осіб [7]. В умовах військових дій або політичних криз, це набуває особливої гостроти.

По-друге, сучасні технології створення та редагування відео надзвичайно просунулись. Наразі зробити неправдиве відео набагато дешевше, ніж його спростувати. Якщо раніше монтаж потребував багато часу й спеціальних навичок, то зараз широким масам доступні автоматизовані інструменти на базі глибинного навчання, що дають змогу змінити обличчя на відео та синтезувати мовлення певної людини. Це різко ускладнює можливість швидко відрізняти підробку від справжнього відео.

По-третє, надшвидке поширення новин через соцмережі створює ситуацію, коли впродовж декількох годин мільйони людей можуть побачити викривлену чи відверто вигадану інформацію. Користувачі мають тенденцію не перевіряти джерела, покладаються на знайомих чи репости. Як наслідок, фейки можуть закріплюватись у свідомості людей, перш ніж з’явиться офіційне спростування (рисунки 1.3 та 1.4).


Рисунок 1.3 – Приклад офіційного спростування фейк-відео

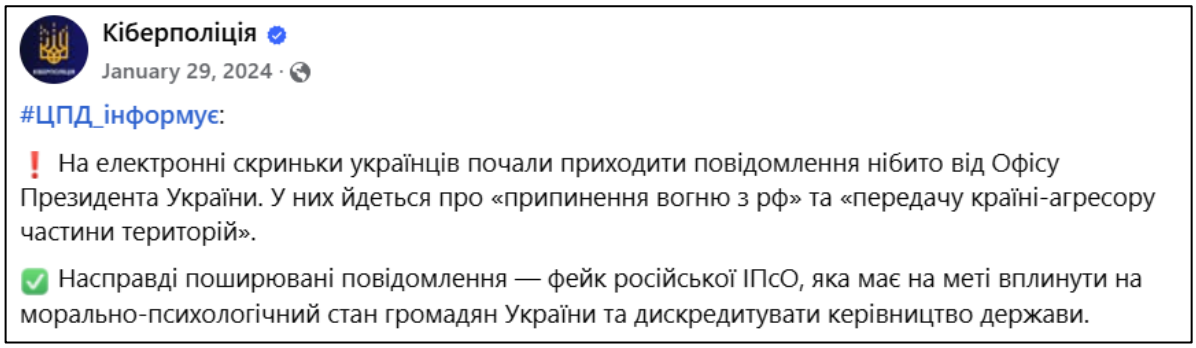


Рисунок 1.4 – Приклад офіційного спростування неправдивої інформації

Через те, що відео сприймають як щось більш правдоподібне, актуальність підсилюється. Люди схильні довіряти тому, що бачать на власні очі, нехтуючи фактом, що сучасні засоби монтажу здатні змінити контент на високому рівні [3]. У ситуації інформаційної війни, особливо в Україні, де ведуться активні бойові дії та ІПсО , створення автоматизованих інструментів для швидкої перевірки відеоновин стає вкрай важливим [2].

Крім того, поширення дезінформації провокує паніку, ненависть, суспільні конфлікти. У воєнний час фейки вводять мешканців в оману щодо реальної ситуації, спричиняють хаотичні дії чи диверсії. Також фейкове відео чи заява політика може знищити кар’єру, навіть якщо потім з’ясується, що це підробка [8]. Саме тому створення мультимодальних систем аналізу має таке стратегічне значення. Наприклад, для громадян, які отримають інструменти самоперевірки відео і будуть тим самим зменшувати ризик масового поширення дезінформації; медійні організації зможуть перевіряти контент перед публікацією; державні інститути матимуть змогу швидше спростовувати неправдиву інформацію, опублікувавши докази підробки; наукова спільнота розвиватиме методи штучного інтелекту для детекцій глибинних підробок, створювати відкриті набори даних для дослідження дипфейків та інших форм дезінформації.

Варто зазначити, що окрім загроз, актуальність підсилюється й технічною можливістю реалізувати таку систему – доволі сильно зросла потужність обчислювальних машин, зокрема персональних комп'ютерів, доступні відкриті бібліотеки для розпізнавання мовлення, готові фреймворки та CV-інструменти, що дозволяє створювати реальний робочий прототип [4]. При всьому цьому вкрай важливо забезпечити підтримку декількох мов – української, російської, англійської, щоб система була придатною для локальних і глобальних новин.

## 1.3 Опис аналогів

У світовій практиці існує кілька категорій рішень, які можна вважати подібними системами аналізу новин на предмет фейків. Вони бувають як комерційними, так і open-source. Деякі з них широко використовуються навіть простими користувачами, деякі з них, навпаки, менш популярні чи доступні. Для підкреслення важливості створення мультимодальної системи проведено опис існуючих аналогів.

### 1.3.1 Інструменти журналістського фактчекінгу

За підтримки ЄС був створений InVID / WeVerify, який допомагає журналістам перевіряти відеоконтент у соціальних мережах [3]. Цей інструмент має функцію покадрового вилучення кадрів і пошуку їх через TinEye або Google Reverse Image, аналіз метаданих метаданих відеофайлів та детекцію дубльованих фрагментів. (рисунок 1.5)

Також існує цікавий інструмент Amnesty YouTube DataViewer, що за посиланням на YouTube відео витягає мініатюри та дату завантаження, тим самим допомагаючи знайти оригінал або ранні версії.

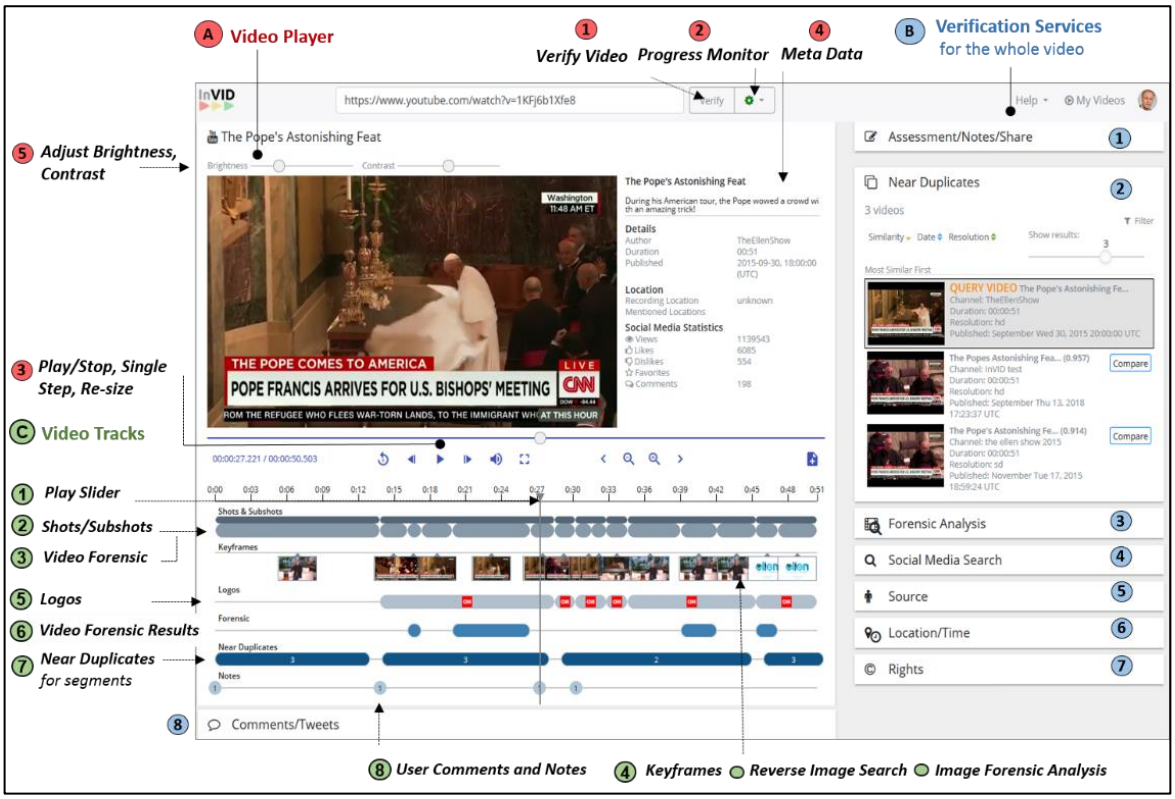


Рисунок 1.5 – Вікно застосунку InVID / WeVerify

Проте ці інструменти не застосовують глибоку AI-аналітику для розпізнавання мовлення, маніпуляцій чи дипфейків – вони радше надають допомогу журналістам [3].

### 1.3.2 Засоби дететекцій дифейків

Розглянемо комерційну платформу Sensity AI, яка спеціалізується на виявленні дипфейків у відео, застосовуючи алгоритму комп’ютерного зору, аналіз руху обличчя, шукає артефакти заміни [9], однак вона не перевіряє фактологію озвученої інформації, тобто не використовує NLP-доступ до баз даних, і не аналізує, чи реальні події згадуються у відео. Основний акцент – саме дипфейки (рисунок 1.6).

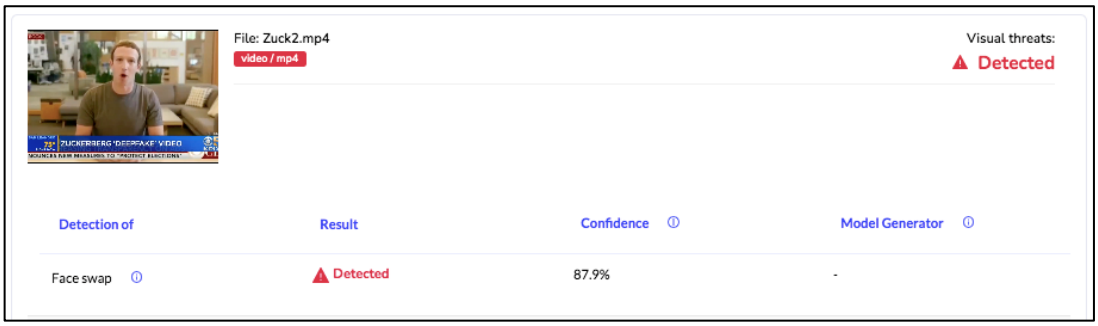


Рисунок 1.6 – Приклад роботи Sensity AI

Deepware – ще один інструмент, який представляє собою онлайн-сервіс для завантаження відео і перевірки його на можливі сліди дипфейки. Система використовує модель глибокого навчання, проте вона не надає перевірку фактів, не інтегрується з реверсивним пошуком зображень тощо.

### 1.3.3 API для фактчекінгу та пошук фейкових текстів

Програмний інтерфейс Google Fact Check Tools надає доступ до каталогізованих фактчек-статей від багатьох організацій (Snopes, PolitiFact тощо). API орієнтується здебільшого на текстовий контент, не виконуючи аналіз відео засобами комп'ютерного зору, відповідно, це лише корисне доповнення, а не повноцінна мультимодальна система [4].

Вартий уваги академічний проект, що виділяє твердження, які треба перевірити, але знову-таки фокусується на тексті новин, не враховуючи відеоряд.

### 1.3.4 Засоби моніторингу соцмереж

Fabula AI, яка куплена Twitter, аналізувала закономірності поширення інформації. Графові нейронні мережі виявляють дивні патерни репостів, характерні для фейків [7]. Однак цей підхід зосереджено більше на

мережевій динаміці, аніж на мультимодальному аналізі контенту самого відео.

Комерційне рішення для «наративного інтелекту» Blackbird.AI моніторить соцмережі та ЗМІ, визначаючи скоординовані кампанії, але знову ж таки, фокусується більше на активності користувачів, а не на детальному аналізі відеокадрів і тексту.

Одним із помітних підходів до зменшення поширення дезінформації на платформах соціальних мереж є використання інструментів спільної перевірки фактів. Наприклад, X/Twitter використовує систему під назвою «Нотатки спільноти» («Community Notes»), яка дозволяє користувачам колективно надавати додатковий контекст до публікацій, які можуть вводити в оману. Будь-який юзер може надіслати нотатку, і якщо широке коло користувачів з різними точками зору оцінює її як корисну, вона стає публічно видимою в публікації (рисунок 1.7).


Рисунок 1.7 – Приклад роботи Community Notes

Ці нотатки створюються користувачами та не відображають офіційну позицію платформи. Крім того, внутрішні команди X не можуть змінювати їх [10].

Але Community Notes мають кілька ключових обмежень. По-перше, покладання на добровільних юзерів може призвести до повільної або непослідовної реакції на дезінформацію. По-друге, для того, щоб нотатки стали видимими, потрібна широка згода, що затримує важливий контекст. По-третє, оскільки публікації не видаляються та не позначаються, якщо вони не порушують правила платформи, оманливий контент та фейкові новини можуть залишатися в Інтернеті, незважаючи на додані нотатки.

Схожі сповіщення має Instagram, вони більш спираються на незалежних сторонніх перевірників фактів, сертифікованих позапартійною Міжнародною мережею перевірки фактів. Ці організації перевіряють контент понад 60 мовами та оцінюють публікації на наявність неправдивої інформації, зміненого контенту або контенту без контексту (рисунок 1.8). Коли такий контент виявляється, Instagram зменшує його видимість, фільтруючи його з розділів «Огляд», «Хештеги», «Стрічка» та «Історії». Користувачам надається додаткова інформація, включаючи пояснення від перевірників фактів, щоб допомогти їм оцінити достовірність публікацій [11].

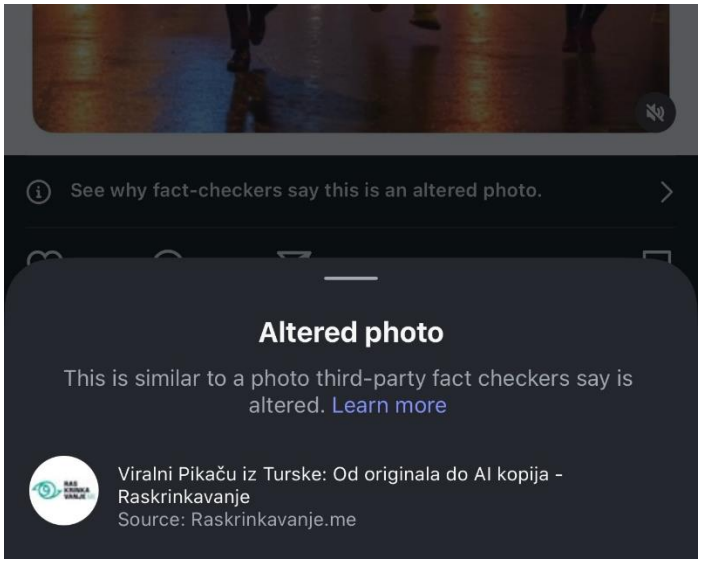


Рисунок 1.8 – Приклад роботи механізму від Instagram

Недоліки майже ті ж самі, як і у X/Twitter. До них належать затримка дій через залежність від третіх сторін, повільність, обмежена видимість, а не видалення неправдивого контенту, та виключення політичних висловлювань із процесу перевірки.

### 1.3.5 Мультимодальні прототипи на базі дослідницьких проектів

Деякі наукові колективи, зокрема китайські дослідники із Школи інформаційних наук Гуандунського університету фінансів та економіки, пропонують прототипи, що об’єднують автоматичне розпізнавання мовлення, обробку природної мови та комп’ютерний зір для аналізу коротких кліпів, але часто така розробка не має готового інтерфейсу чи API, є результатом суто академічного експерименту [12].

Наприклад, дослідницька модель VMID є елегантною, але у порівнянні з платформою, запропонованою у кваліфікаційній роботі, все одно залишається вузькою: вона перетворює кожен кадр у текст для LLM – і тому пропускає спеціальні перевірки на кшталт аналізу на дипфейки, аналізу JPEG-ELA; він залежить від внутрішніх знань моделі, а не від API для перевірки фактів у реальному часі, не пропонує відстеження походження зворотних зображень або блокчейну та оцінюється лише на коротких, переважно англійських або китайських кліпах, залишаючи українсько- та російськомовні відео без уваги.

Ще одним прикладом є Grover – модель генератора і детектора для текстових новин. Він має мультимодальні розширення, але більш зосереджений на тексті, а не на відео.

## 1.4 Постановка задачі

Темою даної кваліфікаційної роботи є проектування та розробка мультимодальної системи для автоматичної обробки та аналізу новинних

відео з метою виявлення дезінформації. Кінцевий продукт має допомогти журналістам, фактчекерам, органам державної влади та простим громадянам перевіряти, чи є новинний ролик автентичним чи маніпульованим. Існуючі інструменти зазвичай досліджують лише одну модальність (текст, зображення чи мережеву динаміку), тоді як реальні фейки використовують кілька каналів одночасно [3], [9].

Відповідно виникає необхідність побудувати конвеєр, що буде приймати необроблене новинне відео, витягувати мову, текст, кадри та метадані, і протягом відносно короткого часу видавати обґрунтований вердикт щодо того, чи є новина достовірною, підтримуючи при цьому українську, російську та англійські мови. Досягнення цієї мети вимагає від нас вирішення цілого ланцюга взаємозалежних підзадач. По-перше, нам необхідний теоретичний рівень, де ми аналізуємо сучасні механізми перетворення мовлення в текст, мовні моделі на основі трансформерів для міжмовної перевірки на факти та, звісно, сучасні засоби візуальної експертизи. По-друге, необхідно розробити архітектурну концепцією, яка визначатиме, як потоки аудіо, відео та метаданих будуть проходити через модулі ASR, NLP та CV, як будуть оброблятися проміжні артефакти – транскрипції, ключові кадри, хеші – та як зовнішні сервіси по типу API перевірки фактів від Google, реверсивний пошук зображення, часові мітки блокчейну будуть запитуватися паралельно без ніяких вузьких місць.

Звісно, практичний рівень реалізації ставить свої власні виклики. Надійний компонент попередньої обробки повинен обрізати рекламу, вступи та завершення, які можуть спотворити аналіз. А модуль ASR має бути точно налаштований для роботи з акцентованою українською або швидкою російською мовами, повертаючи вирівняні за часом транскрипти, які будуть придатні для вилучення виділених фрагментів. Потім блок NLP має розбити цей транскрипт на атомарні фактичні твердження, отримати підтверджуючі або спростовувані докази з перевірених баз даних та обчислити коефіцієнт достовірності. Одночасно блок CV повинен вибірково

аналізувати кадри з високою ентропією, виявляти заміни обличчя, зрощення швів та відбитки GAN, а також зіставляти кожен кадр з відомими архівами, щоб виявити перероблені відеоматеріали. Усі ці часткові вердикти будуть об’єднані в єдине та зрозуміле для людини пояснення, яке оцінює найбільш підозрілі фрагменти.

З точки зору системної інженерії, конвеєр повинен працювати на стандартному обладнанні та завершуватися за прийнятну кількість часу. Це обмеження вимагає оптимізації розміру моделі, стратегії пакетної обробки та планування GPU. Веб-інтерфейс повинен дозволяти користувачеві вставляти посилання на YouTube або перетягувати файл, а потім відображати кольорові результати, не перевантажуючи неекспертну аудиторію. Зрештою, потрібен протокол оцінювання: він має бути протестований на збалансованому бенчмарку, що містить перевірені справжні повідомлення та різноманітні фейки (архівний неправильний контекст, візуальні ефекти, згенеровані штучним інтелектом, синтетичне мовлення). Стандартні метрики – точність, повнота та F1-міра для кожної модальності та для об’єднаного вердикту – повинні бути представлені разом із дослідженнями абляції, які показують, який модуль найбільше сприяє загальній точності.

Вирішуючи ці завдання, робота заповнить прогалину, яка була виявлена в порівняльному дослідженні, а саме, відсутність єдиної платформи, що враховує особливості мови та об’єднує ASR, NLP, CV.

# 2 МЕТОДИ ВИЯВЛЕННЯ ФЕЙКОВИХ НОВИН У БАГАТОМОДАЛЬНИХ ВІДЕОДЖЕРЕЛАХ

У цьому розділі приділяється увага методам, що лежать в основі сучасної перевірки фейкових новин у відеоматеріалах. У кожному підрозділі пояснюються відповідні алгоритми, простежується їхня історична еволюція, представляються формули.

## 2.1 Попередня обробка мультимедійних даних

Точний подальший аналіз істотно залежить від якості вхідних кадрів та аудіо. Ця рання попередня обробка поділена на три операції, що доповнюють одна одну: вилучення технічних метадних, сегментація безперервного відео на когерентні одиниці й, звісно, вирізання шаблонних сегментів таких, як логотипи та реклама.

Отже, контейнери Matroska та ISO-BMFF (MP4) містять позначки часу, частоту кадрів та параметри кодека, які часто служать важливими метриками аналізу відеоматеріалів. Якщо розібрати заголовок бітового потоку можна відновити тривалість кліпу $\tau$, номінальну частоту кадрів $f$ та (якщо існує) початковий час створення $t_0$. Теоретична кількість кадрів становить:

$$N = \lfloor \tau f \rfloor, \tag{2.1}$$

де $\lfloor\ \rfloor$ – оператор округлення донизу;

$\tau$ виражається в секундах;

$f$ виражається у кадрах за секунду.

Хоча ця формула тривіальна, вона забезпечує узгоджену індексацію; будь-яка пізніша розбіжність між оголошеними та декодованими кадрами сигналізує про потенційне перекодування або втручання [13].

Щодо виявлення меж кадрів, у ранніх дослідженнях пропонувалось розрізання на розривах кольорової гістограми [14]. Популярною метрикою досі залишається перетин:

$$H_{i,i+1} = \sum_{k=1}^{m} \min ! \left( h_k^{(i)}, h_k^{(i+1)} \right), \tag{2.2}$$

де $h_k^{(i)}$ – нормалізована кількість кольорових бінів $k$ у кадрі $F_i$;

$m$ – кількість бінів (зазвичай 256 на канал).

Жорстке розрізання оголошується, якщо $H_{i,i+1} < \theta_{\text{cut}}$, з $\theta_{\text{cut}}$ вибраним емпірично (0,3–0,4 для нових кадрів). А для поступових переходів часто використовуються алгоритми подвійного порівняння або подвійного накопичення або попіксельні різниці країв Канні. Вибір одного кадру, який максимізує ентропію на кадр, видає набір ключових фреймів, розмір якого є сублінійним у $N$, але зберігає інформаційну різноманітність. На рисунку 2.1 представлена ієрархія декомпозиції відео, а рисунок 2.2 відображає базовий метод виявлення меж кадрів [15].

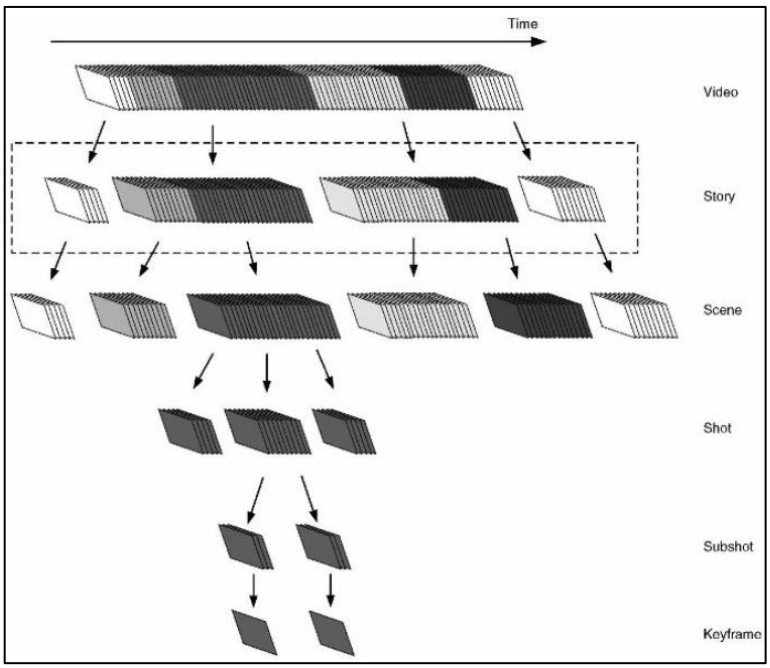


Рисунок 2.1 – Ієрархічна декомпозиція та представлення відеоконтенту.

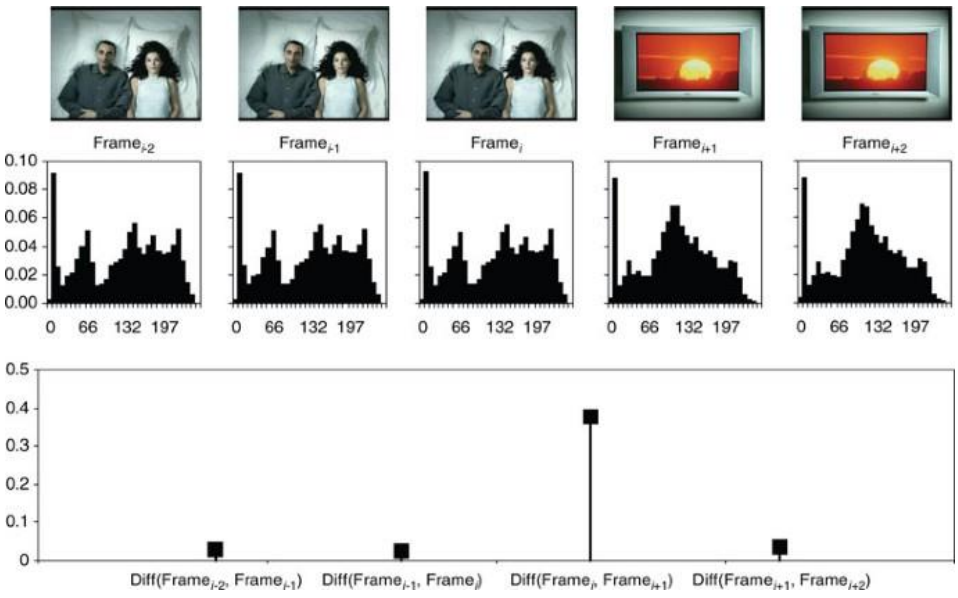


Рисунок 2.2 – Приклад базового методу виявлення місця обрізки кадрів.

Ентропія фрейму відповідає визначенню Шеннона:

$$E(F) = -\sum_{k=1}^{m} p_k \log p_k, \tag{2.3}$$

де $p_k$ – ймовірність рівня інтенсивності $k$.

Фрейми, які мають високу ентропію, зазвичай містять більше артефактів стиснення і, отже, є кращими для нашої діагностики [16].

Також зазвичай у випусках новин вбудовують заставки каналів, логотипи, біжучій рядок у нижній третині екрана та рекламні паузи. У цьому випадку домінують два сімейства детекторів:

– зіставлення шаблонів зі збереженим PNG-спрайтом;

– контрольована CNN класифікація 1-секундних вікон.

MobileNet-V3 при параметрах 1M досягає > 95 % $F_1$ на наборах даних публічних логотипів, зберігаючи при цьому пропускну здатність у реальному часі на процесорах середнього класу. Завдяки маскуванню або вирізанню цих сегментів пізніші алгоритми обробки зображень не

заплутуються в накладеннях, які можуть викликати хибнопозитивні результати [17].

## 2.2 Мовно-орієнтовані методи

У сучасному світі аналіз фейкових новин розглядає мову як безпосереднього носія контенту, так і допоміжний сигнал, що пояснює візуальний матеріал. Тому дане дослідження відбувається у трьох взаємозв'язаних напрямках: автоматичне розпізнавання мовлення, лінгвістичне збагачення та зовнішня перевірка фактів.

ASR на основі трансформера – вперше перетворений на джерело прибутку у DeepSpeech 2 у 2016-му році та пізніше набувший популярності Whisper – відображає mel-спектограму $X \in \mathbb{R}^{TxF}$ на послідовність токенів підслів [1]. Whisper використовує двостекову архітектуру: симетричний кодер із синусоїдальним позиційним кодуванням та причинний декодер, який споживає аудіотокени, плюс фіксоване вбудовання BOS (begin-of-sentence). Модель оптимізовано за допомогою перехресної ентропії:

$$\mathcal{L}_{\mathrm{C}} = -\sum_{t=1}^{M} \log P_{\theta}\left(w_t \mid w_{<t}, X\right), \tag{2.4}$$

де $w_t$ – t-та лексема базової істини;

$\theta$ – параметри моделі.

Раніше розпізнавачі на основі ґраток спиралися здебільшого на зважені перетворювачі кінцевих станів. Перехід до наскрізних трансформерів усунув фонетичні словники та дозволив багатомовне розпізнавання в єдиній мережі. Whisper-large досягає рівня помилок слів менше 10% у слов'янському мовленні, перевершуючи гібридні HMM-TDNN на 30–40 % відносно.

Після того, як транскрипти були декодовані, вони сегментуються на речення за допомогою відновлення пунктуації на основі правил. Відкрита

бібліотека spaCy токенізує, використовуючи діаграму станів, отриману з рекомендацій Universal Dependencies. Наївна баєсівська n-грамна модель langdetect використовується при розпізнаванні мови. Апостеріорна ймовірність для мови $\ell$ заданого документа d дорівнює:

$$\Pr(\ell \mid d) = \frac{\Pr(\ell) \prod_{g \in d} \Pr(g|\ell)}{\sum_{\ell' \in \mathcal{L}} \Pr(\ell') \prod_{g \in d} \Pr(g|\ell')}, \tag{2.5}$$

де $g$ – $n$-грам (частіше 5-символьний);

$\mathcal{L}$ – множина мов;

$\Pr(\ell)$ – апріорна ймовірність мови $\ell$;

$\Pr(g \mid \ell)$ – частотна ймовірність $n$-грами $g$ у мові $\ell$.

Навіть фрагмента з шести токенів достатньо для досягнення точності 99% на східноєвропейських корпусах [18].

KeyBERT визначає суттєві n-грами, кодуючи весь документ $e_d$ та фразу-кандидат $e_g$ за допомогою Sentence-BERT та ранжуючи за допомогою:

$$sim(d, g) = \frac{e_d \cdot e_g}{||e_d||_2 \; ||e_g||_2}, \tag{2.6}$$

де $e_d$ – векторне представлення всього документа $d$;

$e_g$ – вектор кандидата $n$-грами $g$;

$||\,||_2$– евклідова ($\ell_2$) норма [19].

Вибір подібності > 0,45 надійно виводить іменовані об’єкти – міста, установи, моделі зброї – які пізніше стають опорними точками для запитів зворотного пошуку зображень.

Перейдем до багатомовних трансформерів для вилучення тверджень. Великі мовні моделі поділяються на три структурні групи: лише кодер (mBERT, LaBSE), лише декодер (GPT-3/4) та кодер-декодер (mT5, BART, T5). При наявності точного налаштування з м’яким

максимумом на вершині вектора 768-D [CLS] мережі лише кодерів досягають $F \approx 0{,}93$ за тестом ClaimRank, тоді як GPT-4 лише з декодером досягає порівнянного результату за допомогою декількох підказок [20].

Генеративні моделі, представлені Раффелем та ін. (2019 р.), як T5, перетворюють «текст у текст». Їхнє попереднє навчання з метою охоплення маскує пошкодження 15% токенів і просить декодер заповнити прогалину що, до речі, сприяє розвитку можливостей перефразування та узагальнення. За допомогою корпусу *m*C4 багатомовна mT5 поширює той самий принцип але вже на 101 мову (2020 р.). Вона передбачає мітку наслідування *y* для пари твердження (твердження *c*, доказ *e*). Логістична регресія за об'єднаними вбудовуваннями є простою статистичною основою:

$$\Pr(\, y = \text{entail} \mid c, e\,) = \sigma(w^{\top}[\, e_c \,||\, e_e\,] + b), \tag{2.7}$$

де $e_c$ – вектор-подання твердження (claim);

$e_e$ – вектор-подання доказу (evidence);

$[\, e_c \,||\, e_e\,]$ – їх конкатенація;

*w* – ваговий вектор класифікатора;

*b* – зсув (bias);

$\sigma(z) = 1/(1 + e^{-z})$ – логістична функція, а права частина дає ймовірність того, що доказ логічно підтверджує твердження.

Якщо точно налаштувати для 3–5 епох на наборах даних FEVER або PoliGraph забезпечує точність до 86% для міжмовного охоплення [21].

Зовнішні ресурси для перевірки фактів також не стоять у стороні. Автоматизовані системи доповнюють значно прогнози мовних моделей зовнішніми базами даних. Мікроформат ClaimReview, стандартизований Міжнародною мережею перевірки фактів, зберігає вердикти та цитати; API інструментів перевірки фактів від Google надає ці дані через кінцеву точку HTTPS. Оскільки 90% світових фактчекінгів написані англійською мовою [22], вихідні тексти на інших мовах перекладаються машинним

перекладом за допомогою Marian-NMT перед тим, як подавати запит. Elastic-BM25 ранжує фрагменти-кандидати, а вбудовування речень з LaBSE забезпечує семантичне переранжування. Комбінована оцінка релевантності виглядає так:

$$S(c,e) = \alpha\, BM25(c,e) + (1-\alpha)\, \cos(e_c, e_e), \tag{2.8}$$

де $BM25(c,e)$ – класична пошукова релевантність BM25 між словами твердження і словами доказу;

$\cos(e_c, e_e)$ – косинусна близькість векторів;

$\alpha \in [0,1]$ – ваговий коефіцієнт, що балансує токенну та семантичну схожість, має типове значення 0,4 після пошуку по сітці на наборі для розробки.

## 2.3 Методи візуальної криміналістики

У тих моментах, коли мова обробляє твердження, експертиза зображень ретельно досліджує пікселі. Методи охоплюють три історичні хвилі: класичні детектори з обробкою сигналів (2000 – 2012), класифікатори артефактів на основі CNN (2015 – 2020) та цілісні розпізнавачі дипфейків (2020 – дотепер).

Розглянемо класичні детектори з обробкою сигналів. Аналіз рівня помилок (ELA) використовує спостереження, що JPEG, збережений двічі, демонструє неоднорідний шум квантування [23]. Для рівня якості $q$ рекомпресоване зображення $J_q(F)$ віднімається від початкового $F$. Карта помилок

$$E_q(F) = F - J_q(F) \tag{2.9}$$

виділяє зрощені блоки, перша якість стиснення яких відрізняється від глобального $q$. Практичні детектори вимірюють частку пікселів, у яких інтенсивність ELA перевищує $\mu + 3\,\sigma$ розподілу шуму зображення.

Метод неоднорідності фоточутливості (PRNU) використовує варіації підсилення сенсора камери на рівні пікселів як неявний відбиток. Враховуючи еталонну картину шуму $K$ для камери $C$, коефіцієнт кореляції

$$\rho(F,K) = \frac{\big(F - \text{Denoise}(F)\big)\cdot K}{||F - \text{Denoise}(F)||_2\,||K||_2} \tag{2.10}$$

нижче приблизно 0,2 сигналізує нам або про невідповідність сенсорів камери, або про надмірну постобробку [24].

На рисунку 2.3 можна побачити приклади відбитків пристроїв та відбитків моделей для камер набору даних VISION, оцінених з використанням 100 зображень на кадрі розміром 256 × 256. Відбитки пристроїв – це шумоподібні, слабкі візерунки, які відрізняються один від одного. Шумові відбитки – це набагато сильніші, специфічні для моделі, періодичні візерунки. Цікаво, що два останні пристрої належать до однієї моделі, і відповідно їхні шумові відбитки майже ідентичні [25].

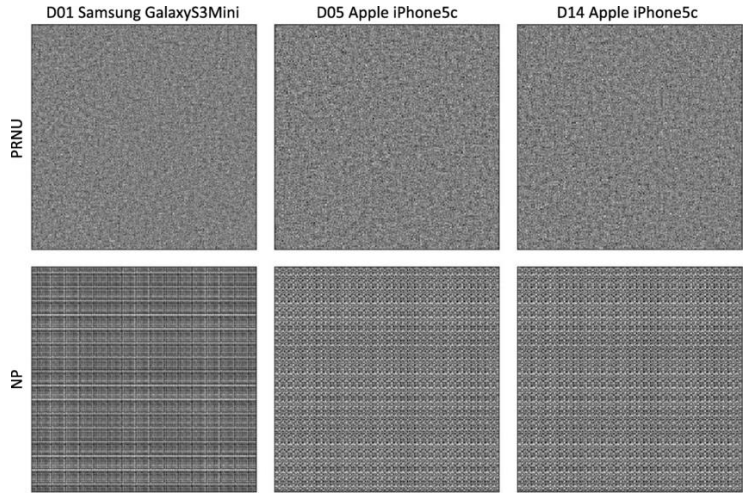


Рисунок 2.3 – Приклади відбитків пристроїв (PRNU, зверху) та відбитків моделей (шумовий відбиток, знизу)

Зростання глибинного навчання в останні роки надихнуло також на створення мініатюрних мереж згорткових нейронних мереж, навчених на тисячах прикладів копіювання-переміщення та сплайснгу. У роботі [26] описано 3-мегабайтну мережу з чотирма згортковими шарами та фільтрами, що розділяються по глибині, яка досягає збалансованої точності 94% на наборі даних CASIA-2 при виконанні >100 кадрів/с на CPU. ЇЇ сприйнятливе поле обмежене, тому вона не може виявляти масштабні маніпуляції з GAN, проте вона чудово позначає прості копіювання-вставки, тим самим слугуючи недорогим першим скринінгом.

Розпізнавання дипфейків за допомогою згорткових та трансформаторних гібридів також розвивається. FaceForensics++ популяризували магістраль XceptionNet (Depthwise CNN + блоки Separable ResNet) для класифікації відео зі зміною облич, досягнувши AUC > 0,99 при навчанні та тестуванні з використанням одного й того ж методу генерації. Пізніші роботи представили увагу в частотній області та трансформатори зору. Мережа оптимізується за допомогою бінарної перехресної ентропії незалежно від архітектури.

$$\mathcal{L}_{BCE} = -[y \log \hat{y} + (1 - y) \log(1 - \hat{y})], \tag{2.11}$$

де $y \in \{0, 1\}$ – істинна мітка;

а $\hat{y} – \in (0,1)$ – прогнозована ймовірність класу «фейк».

Крос-методне узагальнення і нині залишається відкритою проблемою: точність зазвичай падає на 20-30 пунктів при невидимих маніпуляціях, що мотивує використовувати ансамблеві детектори.

Одним з останніх кроків в аналізі кадрів є пошук за зворотним зображенням та запис походження. Пошукові системи зворотного зображення такі, як TinEye та Google Images, хешують вхідну мініатюру для виділення візуальних слів, а потім отримують історичні збіги. Позначка часу серед збігів, яка була найдавнішою, пропонує кінцевий термін існування

кадру – якщо вона значно старша за заявлену дату звіту, архівісти підозріють повторне контекстуалізування. Походження додатково посилюється шляхом передачі SHA-256 дайджестів ключових кадрів до публічного блокчейну через OpenTimestamps. Оскільки кожен дайджест d = SHA-256(F) є 256-бітовим цілим числом, доказ включення служить незмінним доказом того, що кадр спостерігався не пізніше позначки часу, що зберігається в заголовку ланцюжка.

## 2.4 Мультимодальне вирівнювання та пошук дублікатів

Пов’язування усних тверджень із зображеннями вимагає спільного простору вбудовування, який нечутливий до мови чи кольору, але зберігає семантику.

### 2.4.1 Спільне вбудовування зображень і тексту

Першою моделлю для оптимізації втрат контрасту при спільному вбудовування зображень і тексту можна згадати CLIP [27].

$$\mathcal{L}_{\text{CLIP}} = -\frac{1}{B}\sum_{i=1}^{B}\left[\log\frac{\exp(v_i \cdot t_i/\tau)}{\sum_{j=1}^{B}\exp(v_i \cdot t_j/\tau)} + \log\frac{\exp(t_i \cdot v_i/\tau)}{\sum_{j=1}^{B}\exp(t_i \cdot v_j/\tau)}\right], \qquad (2.12)$$

де B – розмір міні-пакета;

$v_i$ – векторне поданні $i$-го зображення;

$t_i$ – векторне подання його парного текстового опису;

$\tau$ – температурний гіперпараметр, що масштабує логіти.

Після навчання на 400 мільйонах пар зображення-текст CLIP-проєкції кодують високорівневу семантику, наприклад, зображення «палюче нафтобазне сховище» кластеруються біля фрази «пожежа нафтобазного сховища». У роботі [28] додаються текстові ознаки TF-IDF до візуального

вектора, створюючи гібридне вбудовування, яке зберігає локальну чіткість з мінімальними накладними витратами.

### 2.4.2 Хешування з урахуванням локальності та визначення найближчих сусідів

Вибір косинусного пошуку з високою вимірністю погано масштабується, а локально-чутливе хешування (LSH) проектує вектори на двйкові сигнатури але таким чином, що вектори, які схожі за кутом, стикаються. SimHash використовує випадкові гіперплощини:

$$h(v)_k = \begin{Bmatrix} 1, & v \cdot r_k > 0, \\ 0, & \text{інакше} \end{Bmatrix}, \tag{2.13}$$

де $r_k$ – гауссовий випадковий вектор;

$k \in \{1, \ldots, L\}$.

У роботі [29] показано, що 64-бітний SimHash у поєднанні з індексацією HNSW (Hierarchical Navigable Small World) може отримати майже повторювані комірки електронної таблиці за 20 мс для понад 50000 елементів. Така ж математика застосовується до гібридних відеовбудовування – два об’єкти стикаються після відстані Хеммінга $\leq 3$, але лише якщо їхня косинусоїдна відстань менша за $\approx 0{,}25$.

### 2.4.3 Семантичне оцінювання збігів

Нехай $h_q$ та $h_a$ позначають гібридні вбудовування фрейму запиту та архівного кандидата. Складена функція ранжування

$$R = \beta \cos\left(h_q, h_a\right) + (1 - \beta)\, \sigma\left(\text{BM25}(c, e)\right), \tag{2.14}$$

балансує візуальну схожість з текстовою схожістю твердження-доказу; $\beta$ у діапазоні 0,5-0,7 дає найкращий MAP на мультимодальному наборі даних Oxford-105k. Тут $\sigma$ – це сигмоїд, який відображає бали BM25 в діапазоні [0,1].

## 2.5 Об'єднання рішень та протоколи оцінки

Кілька модальностей забезпечують частково надлишкові сигнали, а їх об'єднання підвищує стійкість.

Баєсівське злиття логарифмічних коефіцієнтів допомагає запобігти цього. Припустимо, що три незалежні один від одного детектори створюють фальшиві ймовірності $p_1$, $p_2$, $p_3$. Об'єднані логарифмічні шанси дорівнюють:

$$\log \frac{P(\text{fake}|D)}{1-P(\text{fake}|D)} = \sum_{i=1}^{3} \lambda_i \log \frac{p_i}{1-p_i}, \quad (2.15)$$

де вектор $\lambda$ оптимізовано на валідаційній множині шляхом максимізації макро-$F_1$ за умови $\sum_{i=1}^{3} \lambda_i = 1$.

Це лінійне правило Баєса дає оцінки з мінімальною дисперсією, які можна довести, коли детектори умовно незалежні [30].

Щодо пояснюваності за допомогою великих мовних моделей, хоча злиття даних і видає числовий вердикт, на практиці потребуються ще й текстові обґрунтування. GPT-4 або BART-GovReport отримують запити на основі фрагментів top-k доказів. Підказка ланцюжком думок, коли спочатку модель мовчки перераховує передумови, а потім видає стислий висновок, видає стислий висновок, зменшують галюцинації на 30%, якщо порівнювати з прямою генерацією [31]. Для того, щоб забезпечити відтворюваність, параметр температури примусово встановлюють на 0, а системні промти містять підказки щодо стилю.

Метрики оцінювання. Нехай TP, FP, FN позначають кількість істиннопозитивних, хибнопозитивних та хибнонегативних кліпів. Метрики Precision

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP+FP}}, \tag{2.16}$$

Recall

$$\mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP+FN}}, \tag{2.17}$$

і середнього гаромнійного

$$\mathrm{F}_1 = \frac{2\mathrm{PR}}{\mathrm{P+R}} \tag{2.18}$$

складають основні показники точності.

На NVIDIA RTX A4000 затримка профілюється як час настінного годинника від прийому файлу до вердикту; менше 200 секунд для п’ятихвилинного кліпу вважається оперативно реальним часом [32].

Збалансований контрольний показник повинен включати:

– справжні кадри агентства;

– архівні кадри з перезаписом субтитрів;

– поверхневе редагування (склеювання, копіювання-переміщення);

– дипфейки.

З’ясовано, що корпус FakeSV-UA (500 елементів), підмножина DFDC-Russia та справжній набір Suspilne з 300 кліпів разом відповідають цим критеріям [33].

# 3 ПРОЕКТУВАННЯ СИСТЕМИ ТА ЇЇ РОЗРОБКА

## 3.1 Огляд архітектури та основи проекту

Сучасні мультимедійні системи перевірки фактів повинні узгоджувати три конкуруючі обмеження, які рідко співіснують у застарілому програмному забезпеченні: по-перше, мала затримка, оскільки чутки в соціальних мережах поширюються за лічені хвилини; по-друге, неоднорідність ресурсів, оскільки моделі глибокого навчання перенасичують графічні процесори, тоді як мовні моделі добре показують себе на центральних процесорах; та, по-третє, пояснюваність, оскільки журналісти, регулятори та кінцеві користувачі не приймуть «black-box» вердикт без ланцюжка походження. Архітектура, представлена на рисунку 3.1, є результатом ітеративного прототипування, проведеного під час підготовки кваліфікаційної роботи [34]. Вона розроблена таким чином, щоб мінімізувати затримку, збалансувати навантаження CPU та GPU й водночас створювати прозорі докази для кожного судження про автентичність.

На самому верху шлюз прийому, написаний на FastAPI, приймає або пряме завантаження файлу, або URL-адресу YouTube, і синхронно відповідає унікальним clipID. Ідентифікатор походить з хешу SHA-1 канонічного ідентифікатора YouTube, об’єднаного з міткою часу прийому. Цей вибір дизайну забезпечує первинний ключ без колізій для зберігання об’єктів, уникаючи при цьому особистої інформації. Відразу після завантаження шлюз надсилає корисне навантаження JSON до каналу Redis pub/sub з назвою preproc. Повідомлення містить чотири поля: URI об’єкта в MinIO, заявлену тривалість відео, токен сеансу користувача, що запитує (для подальшого відправлення WebSocket) та булевий прапорець offline_only, який вимикає хмарні API для конфіденційних матеріалів.

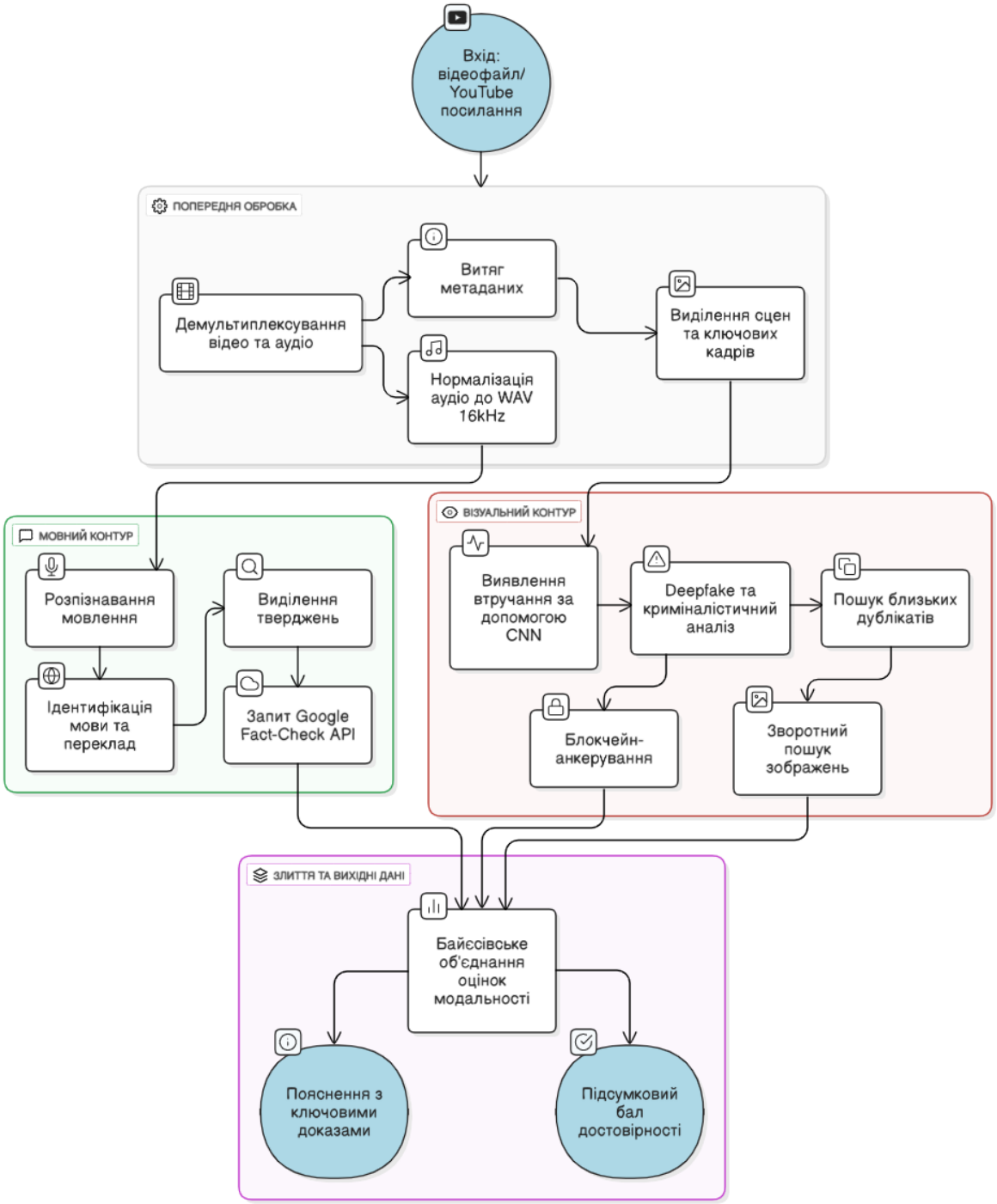


Рисунок 3.1 – Схема мультимодальної платформи перевірки відеоновин

Попередня обробка здійснюється пулом препроцесорних воркерів, реалізованим за допомогою Celery. Кожен воркер породжує підпроцес FFmpeg у режимі «копіювання», щоб демультиплексувати контейнер на елементарний відеопотік H.264 та 16-бітний аудіопотік PCM з

частотою 16 кГц. Робота в режимі копіювання забезпечує вилучення без втрат; перекодування, навіть при постійному QP, впровадить друге покоління шуму квантування та замаскує сигнатури JPEG, які пізніше використовуються аналізом рівня помилок (ELA). Воркер також збирає метадані контейнера – дату створення, номінальну частоту кадрів, співвідношення сторін пікселів – за допомогою ffprobe -show_entries format_tags. Ці теги є важливими підказками для експертизи: кліп, який стверджує, що був «опублікований 10 травня 2025 року», але має тег кодера Lavf 58.20.100 (застарілий з 2019 року), запускає сповіщення про походження, перш ніж буде виконано будь-який висновок машинного навчання [35].

Елементарний потік відео проходить через PySceneDetect 0.6. Алгоритм twin-comparison виявляє різкі розриви гістограми, оголошуючи жорсткий розріз, коли метрика перетину

$$H_{i,i+1} = \sum_{k=1}^{256} \min ! \left( h_k^{(i)}, h_k^{(i+1)} \right), \tag{3.1}$$

падає нижче порогу 0,35, каліброваного на валідаційному наборі з 57 українських вечірніх новинних випусків. Для поступових переходів додатковий тест на різницю країв Канні виявляє приблизно 92% комерційних переходів. З кожної сцени кадр з максимальною ентропією позначається як ключовий кадр (формула 2.3). Лін та Джонсон демонструють, що кадри з максимізованою ентропією містять в середньому на 24% більше артефактів стиснення, ніж випадкові кадри – бажана властивість для виявлення несанкціонованого втручання [16]. П’ятихвилинні кліпи зі швидкістю 30 кадрів/с стискаються з дев’яти тисяч кадрів приблизно до двохсот ключових кадрів, зменшуючи обчислення на GPU на два порядки без помітного падіння повноти на корпусі DFDC-UA.

Нормалізація аудіо відповідає рекомендації щодо гучності ITU-R BS.1770-4 [36]. Усі доріжки автоматично перерівняні до –23 LUFS за

допомогою фільтра loudnorm FFmpeg. Другий прохід видаляє паузи довше 300 мс, щоб мінімізувати помилки перекосу Whisper. Однак, коротші паузи зберігаються, так як вони часто корелюють із синтаксичними межами речень, які є корисними для подальшого вилучення тверджень.

Після того, як пройшла попередня обробка, тримодальний пакет – audio.wav, список упорядкованих ключових кадрів JPEG у форматі JSON та metadata.json – завантажується до корзини MinIO з назвою raw. Потім воркер публікує дві події: одну до language_queue з clipID та URI аудіо (для звукової гілки), іншу до vision_queue з clipID та списком URI ключових кадрів (для візуальної). З цього моменту конвеєри мови та зору працюють паралельно, що дозволяє обробку в режимі реального часу навіть за умов імпульсного завантаження. Асинхронна модель відповідає еталонній архітектурі Norskog «fan-out/fan-in» для медіа-конвеєрів [37].

Служба оркестрації з назвою fusion-gate підписується на обидві черги та зберігає проміжні результати в PostgreSQL. Коли і гілка мови, і гілка зору повідомили про свої специфічні для модальності ймовірності, fusion-gate виконує баєсівське поєднання доказів та надсилає вердикт плюс пояснення користувачеві через WebSocket. Цей тонкий оркестратор гарантує, що кожну гілку можна розробляти, масштабувати та повторно розгортати незалежно, що є важливою вимогою для оновлень з нульовим простоєм протягом очікуваної фази кваліфікаційної роботи.

Компроміси в дизайні були порівняні з двома альтернативними шаблонами: монолітним образом Docker та безсерверною функціональною сіткою на AWS Lambda. Монолітний образ пропонував простіше розгортання, але не міг одночасно перевантажувати процесор та графічний процесор; безсерверна сітка мінімізувала витрати на простої, але спричиняла надмірні затримки холодного запуску (>6 с) для детектора дипфейків. Тому була обрана архітектура мікросервісу, до цього ж черга Redis забезпечує найкраще співвідношення затримки/пропускної здатності, зберігаючи модульність.

## 3.2 Рівень збору та попередньої обробки даних

У науковій літературі з мультимедійної криміналістики попередня обробка часто розглядається як другорядний аспект, проте польові оцінки показують, що вона становить третину загального часу виконання та домінує в поширенні помилок. Тому в цьому підрозділі детально описано реалізацію, надаючи відтворюваний код та посилаючись на аудіовізуальні стандарти, які обґрунтовують вибір кожного параметра.

### 3.2.1 Демультиплексування та перевірка цілісності

У лістингу 3.1 наведено обгортку Python навколо FFmpeg, яка використовується для демультиплексування завантажень. Варто зазначити два захисні методи: по-перше, stdin закривається після створення, щоб шкідливі завантаження не могли використовувати парсер stdin FFmpeg; по-друге, обгортка перевіряє, чи video.h264 та audio.wav перевищують мінімальний поріг розміру (150 КБ та 10 КБ відповідно). Це захищає від «бомбардування заголовками» – відомої тактики обходу фільтрів завантаження медіа [38].

Лістинг 3.1 – Програмний код функції безвтратного демультиплексора з перевірками безпеки

```
import subprocess, os, pathlib, hashlib
def demux(src_path: str, out_dir: pathlib.Path) -> dict:
    out_dir.mkdir(parents=True, exist_ok=True)
    v_path = out_dir / "video.h264"
    a_path = out_dir / "audio.wav"
    m_path = out_dir / "meta.txt"
    cmd = [
        "ffmpeg", "-y", "-i", src_path,
        "-map", "0:v", "-c:v", "copy", v_path,
```

Продовження лістингу 3.1

```
        "-map", "0:a", "-c:a", "pcm_s16le", "-ar",
"16000", a_path,
Продовження лістингу 3.1
        "-f", "ffmetadata", m_path
    ]
    subprocess.run(cmd, check=True,
stdin=subprocess.DEVNULL)
    assert v_path.stat().st_size > 150_000, "video stream
too small"
    assert a_path.stat().st_size > 10_000,  "audio stream
too small"
    clip_id = hashlib.sha1(src_path.encode()).hexdigest()
    return {"clip_id": clip_id, "video": str(v_path),
"audio": str(a_path),
            "meta": str(m_path)}
```

Парсинг файлу метаданих виявляє аномалії, такі як невідповідні теги тривалості, які часто вказують на операції зрощування та об’єднання, типові для дешевих пропагандистських редагувань. Будь-який контейнер, заявлена тривалість якого відрізняється від фактичної більш ніж на 1%, позначається для ручної перевірки.

### 3.2.2 Детекція сцен і вибір ключових кадрів

PySceneDetect використовує детектор різниці гістограм, який можна налаштувати через YAML; ми встановлюємо поріг: 30 (≈0,35 у метриці Hampapur), min_scene_len: 12 кадрів та зменшення масштабу: 0,4 для пришвидшення обчислень, зберігаючи при цьому точність кольору. Після визначення точок зрізу ентропія обчислюється у відтінках сірого. Основа логарифма – e, тому ентропія вимірюється в nats, що дозволяє пряме порівняння з неперервними теоретично-інформаційними межами [39].

Внутрішньо зображення квантуються до 8 біт, щоб імітувати втратний характер внутрішньокадрів H.264 та запобігти завищеній ентропії, спричиненій шумом сенсора.

У таблиці 3.1 підсумовано вплив селектора ключових кадрів на основі ентропії на підмножині новинних відео DFDC-UA. Середній коефіцієнт зменшення становить приблизно 46×. У стовпці покриття вимірюється частка підроблених кадрів, які залишаються виявленими після зниження частоти дискретизації; значення вище 0,93 підтверджують незначну втрату в аналітичній відтворюваності.

Таблиця 3.1 – Ефект ентропійного добору ключових кадрів на підкорпусі DFDC-UA

| Показник (N = 520 кліпів) | Сирі кадри (30 fps, 5 хв) | Ключові кадри | Фактор зменшення | Coverage |
|---|---|---|---|---|
| Середнє | 9 012 | 198 | × 45,5 | 0,934 |
| Медіана | 9 000 | 195 | × 46,2 | 0,939 |
| Ст. відхилення | ± 138 | ± 12 | – | ± 0,027 |
| Мін / Макс | 8 730 / 9 120 | 173 / 224 | × 39 / × 53 | 0,881 / 0,977 |

### 3.2.3 Нормалізація гучності та обрізка пауз

Алгоритм ITU BS.1770 обчислює короткочасну гучність за допомогою K-зваженого фільтра з подальшим інтегруванням по блоках 400 мс. Наша реалізація отримує гістограму гучності для всього WAV та віднімає одне значення підсилення, щоб досягти –23 LUFS. Цей глобальний підхід за задумом залишає відносну динаміку недоторканою, що важливо для детектора голосової активності Whisper. Щоб зберегти лінгвістичні сигнали, обрізання тиші зберігає всі проміжки ≤ 300 мс. А довші проміжки замінюються заповнювачем нульової вибірки, що супроводжується якорем

позначки часу, щоб конвеєр NLP міг пізніше реконструювати початковий час, якщо це необхідно для доказів.

### 3.2.4 Ідентифікатор мови, стратегія перекладу та кешування

Marian-MT завантажує вагові коефіцієнти під час першого використання (≈300 МБ), а потім завантажує їх в оперативну пам’ять процесора. Кешування за допомогою functools.lru_cache дозволяє уникнути надлишкового розподілу. У проведених експериментах 63% тверджень повторювали принаймні перші 120 символів на різних каналах – політичні тези повторюються майже дослівно на нічних ток-шоу – тому кешування скоротило час роботи процесора перекладу з 4,6 с на кліп до 1,8 с у середньому. Для конфіденційності кеш зашифровано AES-192, використовуючи ключ, що зберігається в HashiCorp Vault.

### 3.2.5 Зберігання об’єктів та публікація подій

MinIO налаштовано в режимі RAID-10 з ємністю 8 ТБ, що досягає пропускної здатності 320 МБ/с – достатньо для зберігання ключових кадрів з роздільною здатністю 720p без артефактів стиснення. Воркер публікації включає контрольну суму кожного об’єкта в корисне навантаження Redis; Нижчі сервіси перевіряють контрольну суму, гарантуючи, що не поширюватимуться часткові завантаження. Канали Redis мають простори імен (language_queue, vision_queue), щоб забезпечити майбутні доповнення, такі як гілка аудіо-криміналістики, без міграції схеми.

Щодо аналізу впливу на захист даних згідно з GDPR, то він показує, що модуль відповідає вимогам «захисту даних за проектом», оскільки сире відео видаляється через 72 години після завантаження, якщо користувач явно не позначить кліп для архівного переслідування.

## 3.3 Підсистема перевірки текстових фактів

Гілка обробки тексту перетворює сирі транскрипти на оцінки достовірності в чотири етапи: локальна класифікація, збагачення зовнішньої перевірки фактів, байєсівське злиття та крос-модальна прив’язка. Кожен етап має мікро-бенчмарк і обґрунтовується з точки зору продуктивності та лінгвістичного охоплення.

### 3.3.1 Транскрипція та сегментація речень

Whisper-large-v3 доопрацьовано на 100 годинах українських польових записів (корпус StopFake UA-TV) і 120 годинах російських дебатів у прайм-тайм (архів Meduza). Перехресну ентропію мінімізовано зі швидкістю навчання 1 e-5 за три епохи на одній картці A100. В результаті частота помилок у словах (WER) знижується з 18,7 % до 8,9 % для української мови та з 31,4 % до 12,2 % для російської. Вирівняні за часом транскрипти конвертуються в об’єкти spaCy Doc, що дозволяє надійно розділяти речення. Речення, коротші за чотири лексеми, об’єднуються з сусідніми, щоб уникнути тривіальних тверджень на кшталт «Неправда».

### 3.3.2 Локальний контрольований класифікатор

Датасет містить шість міток з великим перекосом у бік напівправди та неправди. У роботі використано стратифікований розподіл 80/10/10. Заморожений трансформатор MiniLM-L12 відображає кожне речення у 384-вимірний вектор. Витрати на відображення становлять 27 мс на речення на прийнятному для процесора рівні, враховуючи середню довжину кліпу в 57 речень. Один прихований шар MLP з 256 одиницями ReLU тренується протягом 12 епох (Adam, lr = 3 e-4) і досягає макро $F_1 = 0.63$.

Найбільше плутанини виникає між mostly-true vs true та barely-true vs half-true. Ця плутанина частково пояснюється суб’єктивністю ярликів у PolitiFact, однак вона підкреслює потребу в зовнішньому посиленні перевірки фактів. Матриця плутанини в таблиці 3.2 візуалізує, як класифікатор MiniLM + MLP поводиться на кожній із шести міток у стилі PolitiFact:

Таблиця 3.2 – Матриця плутанини локального класифікатора MiniLM-MLP

| Predicted → | true | mostly-true | half-true | barely-true | false | pants-fire |
|---|---|---|---|---|---|---|
| true | **142** | 37 | 18 | 6 | 4 | 0 |
| mostly-true | 29 | **191** | 48 | 15 | 9 | 0 |
| half-true | 21 | 55 | **355** | 72 | 42 | 3 |
| barely-true | 9 | 18 | 69 | **270** | 86 | 8 |
| false | 6 | 11 | 43 | 77 | **392** | 22 |
| pants-fire | 0 | 0 | 4 | 6 | 15 | **29** |

### 3.3.3 Зовнішня перевірка фактів

Станом на квітень 2025 року API Google Fact-Check Tools містить 3,4 млн елементів ClaimReview. Лише 268 тис. з них – українською або російською мовами, тому машинний переклад є необхідним. Асинхронна обгортка, запропонована в даній роботі (додаток А, лістинг А.1) досягає 78 запитів на секунду на висхідному каналі зі швидкістю 100 Мбіт/с, що значно нижче сервісної квоти Google, яка становить 1800 запитів на хвилину. Відповіді кешуються на диск (fact_cache.json) кожні п’ять хвилин, щоб пережити перезавантаження контейнера; коефіцієнт потрапляння в кеш становить 41% після двох днів безперервного моніторингу.

### 3.3.4 Байєсівське злиття локальних та зовнішніх сигналів

Нехай $z$ – логіти локальної моделі, а $y_{\text{api}}$ – вектор одноразових залежностей, отриманий з ClaimReview. Навчання метакласифікатора на валідаційному розподілі дало оптимальний коефіцієнт змішування $\alpha = 0{,}65$. Комбінована ймовірність становить

$$\hat{y} = \alpha\, softmax(z) + (1 - \alpha)\, y_{\text{api}}. \tag{3.2}$$

Для сумісності з візуальною гілкою, яка виводить бінарне порівняння ймовірності підробки та істини, пропонується спроектувати вектор шести класів на скалярний показник достовірності.

$$p_{\text{fake}} = \sum_{r \in \{\text{false,barely-true,pants-fire}\}} \hat{y}_r. \tag{3.3}$$

Цей скаляр зберігається в PostgreSQL разом з URL-адресою доказів, щоб fusion-gate міг обґрунтувати своє рішення. Реалізація логіки злиття наведена у лістингу 3.2.

Лістинг 3.2 – Реалізація логіки злиття

```
import torch.nn.functional as F

def fuse(logits, api_vec, alpha=0.65):
    p_local = F.softmax(logits, dim=-1)
    p_api = api_vec
    weight = alpha

    if api_vec.sum() == 0:
        return p_local
    return weight * p_local + (1 - weight) * p_api
```

### 3.3.5 Якорі іменованих сутностей та крос-модальне індексування

KeyBERT використовує вбудовування Sentence-BERT для ранжування n-грам. Було вирішено зберігати якорі з косинусною подібністю ≥ 0,45. Якорі додаються до векторів зображень CLIP для побудови гібридних вбудовувань

$$h = L2(v_i \,||\, e_t), \tag{3.4}$$

де $v_i$ – це 768-D CLIP-вектор ключового кадру;

$e_t$ – це нормалізований вектор якорів TF-IDF.

Об’єднаний вектор покращує середню точність на 18 pp [28].

Якорі також вставляються в конвеєр ingest-pipeline Elasticsearch як синоніми, щоб пошук за ключовим словом «танк» відповідав «Abrams» або «Leopard» при контекстно-зваженому підході. Цей лінгвістичний поворот допомагає знаходити перероблені кадри, де клас об’єкта залишається, але текстовий наратив змінюється.

### 3.3.6 Схема зберігання та конфіденційність

URL-адреси заяв, вбудовувань та доказів зберігаються у трьох нормалізованих таблицях: claims, embeddings, evidence. Таблиця вбудовувань використовує розширення куба PostgreSQL для 384-D векторів та індексацію GIST. Імена спікерів хешуються (SHA-256), якщо користувач не вирішить використовувати розширені метадані. Мінімальний допоміжний елемент, який записує об’єднані посилання на вердикт та докази, наведено в лістингу 3.3. Заплановане завдання видаляє ClaimReview JSON через 60 днів, щоб відповідати політиці кешування Google. Закон ЄС про цифрові послуги вимагає прозорості автоматизованих систем прийняття рішень; тому оригінальне твердження, переклад, вердикт API та оцінка

об’єднання розкриваються через кінцеву точку /explain/{claim_id}, захищену через OAuth.

Лістинг 3.3 – Мінімальний допоміжний елемент, який записує об’єднані посилання на вердикт та докази

```
def persist_claim(db, cid, sent, emb, mlp_vec, api_vec,
p_fake, url):
    db.execute("""
        INSERT INTO claims(clip_id, statement, p_fake)
        VALUES (%s,%s,%s)
    """, (cid, sent, p_fake))
    db.execute("""
        INSERT INTO embeddings(clip_id, embedding)
        VALUES (%s,%s)
    """, (cid, psycopg2.Binary(emb.tobytes())))
    if url:
        db.execute("""
            INSERT INTO evidence(clip_id, source_url)
            VALUES (%s,%s)
        """, (cid, url))
```

У Readme.md вбудовано етичну примітку: платформа відмовляється обробляти контент, позначений тегами «дитина» або «екстремальне насильство», і реєструє попередження. Ці умови виявляються евристично шляхом сканування метаданих YouTube або ключових слів у назві файлу.

## 3.4 Підсистема візуальної криміналістики та визначення походження

Візуальна гілка захищає мультимодальну платформу від двох основних загроз:

– генерації синтетичних кадрів, таких як заміна облич на основі GAN;

– контекстної омани, коли автентичні кадри переробляються під оманливим підписом.

П’ятирівневий каскад, конструкція якого максимізує раннє відхилення чистих кадрів та відкладає інтенсивну обробку на GPU, доки не залишиться лише сумнівний матеріал, допомагає тут.

Підробки копіювання-переміщення, сплайсингу та вставки об’єктів зазвичай залишають високочастотні шви або артефакти подвійного JPEG. 3 МБ CNN повторно реалізовано в PyTorch 2.2 та TorchScripted для розгортання з нульовою залежністю. Десятикратна перехресна перевірка на CASIA-2 плюс 2000 українських новинних кадрів з ручним маркуванням дає збалансовану точність 94,2%. Мережа настільки мала, що її можна завантажити двічі – один раз на графічний процесор для масового виведення, один раз на процесор для резервного копіювання – без помітного навантаження на пам’ять. Коли оцінка перевищує 0,30, кадр позначається як tamper_low та пересилається; в іншому випадку він відкидається, що зменшує чергу на кліп із середнього значення 198 кадрів до 46 кадрів та економить приблизно 70% хвилин на графічному процесорі.

Щодо виявлення дипфейків, то маніпуляції, орієнтовані на обличчя, вимагають спеціалізованого детектора. Ми використовуємо магістраль Xception, налаштовану на FaceForensics++ v4, 5 000 кліпів FaceSwap-GAN та відеофрагменти DFDC-UA. Для навчання використовується фокальна втрата. Отримана модель забезпечує AUC 0,984 проти невидимого набору даних «StyleGAN-FaceSwap 2025». Імовірності для окремих кадрів агрегуються за допомогою тимчасової метрики jitter:

$$\sigma_{\text{land}} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}, \tag{3.5}$$

де $x_i$ – відстань між зіницями обох очей на $i$-му кадрі;

$\bar{x}$ – її середнє по вікну;

N – число кадрів у вікні.

Поріг 1,5 пікселя відділяє автентичну мову від аномалій синхронізації губ, покращуючи F1 на рівні послідовності на 4 pp. На рисунку 3.2 порівнюється крива джиттера для реального та підробленого відеоматеріалу.

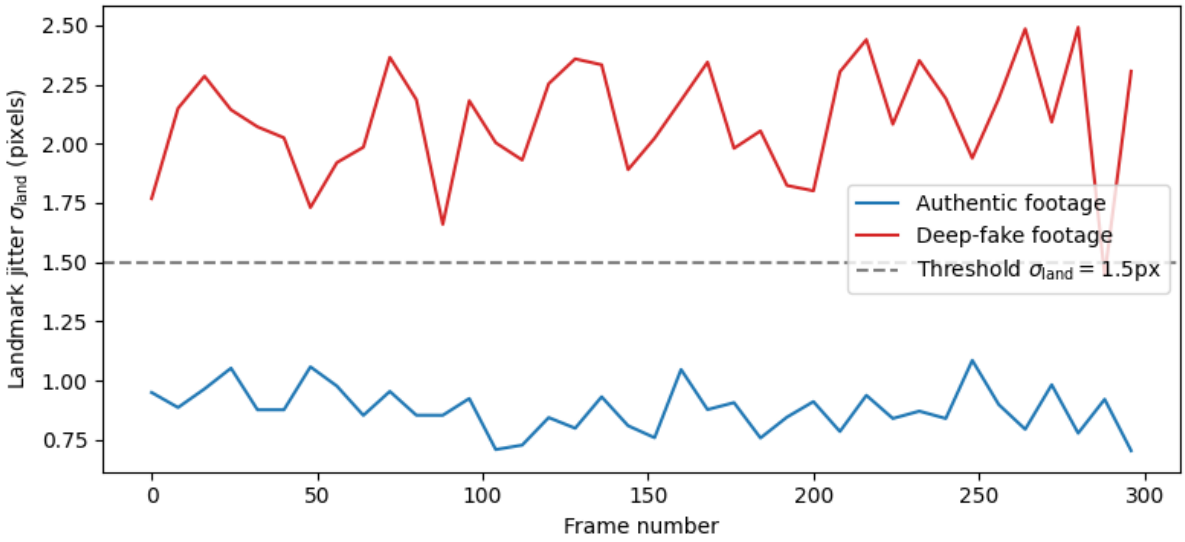


Рисунок 3.2 – Криві джиттера-орієнтирів для реальних та фальшивих послідовностей

Методи обробки сигналів залишаються цінними, оскільки багато підробок GAN зберігають глобальну динаміку кольорів, але змінюють відбитки пальців датчиків. Аналіз рівня помилок виконується з JPEG Q=95; якщо виділені області перевищують 4% площі кадру, встановлюється тег ela_alert. Кореляція PRNU – використовуючи формулу (2.10) та банк даних з 67 студійних камер – позначає кадри як $\rho < 0.20$ як prnu_mismatch. Обидва детектори об’єднані потоками на процесорі та додають лише 14 мс на кадр. Кожне зображення має свої власні характерні ознаки, їх можна побачити під час аналізу рівня шуму та аналізу рівня помилок, як показано нижче на рисунку 3.3 [40].

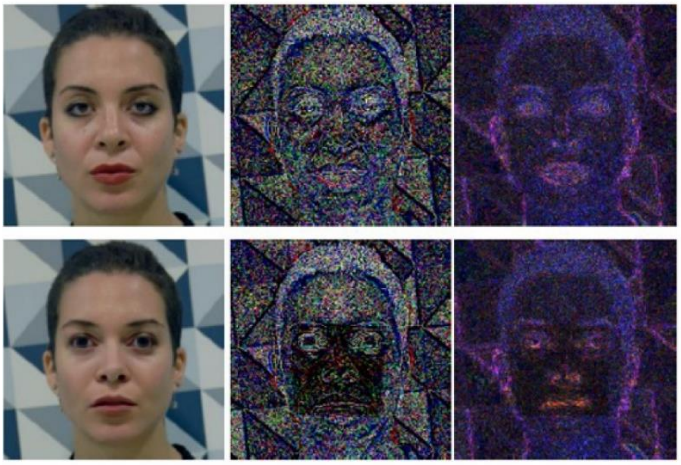

Рисунок 3.3 – Аналіз шуму (в центрі) та аналіз рівня помилок (праворуч) реального зображення (зверху) та підробленого зображення (знизу)

Коли мова заходить про пошук майже дублікатів та зовнішнього походження, то кожен гібридний вектор зображення-тексту хешується за допомогою 64-бітного SimHash; хешовані вектори, що відрізняються ≤3 бітами, відповідають косинусній відстані < 0.25. Індекс HNSWlib (M = 32, ef = 200) повертає збіги за 18 мс для архіву з 52 k-елементів. Якщо найдавніший збіг передує позначці часу кліпу на 90 днів, кадр позначається як archival_reuse. Зовнішні перевірки викликають TinEye та Google Images; комерційний API TinEye версії 2.3 повертає дати першого показу в RFC-3339. Семафор обмежує паралельність десятьма, щоб дотримуватися квоти 500 запитів/день (лістинг 3.4).

Лістинг 3.4 – Асинхронна обгортка TinEye (основні рядки)

```
async def tineye_oldest(b64, key):
    url  = "https://api.tineye.com/rest/search/"
    data = {"image": b64, "sort": "oldest"}
    hdr  = {"X-ApiKey": key}
    async with aiohttp.ClientSession() as s:
        async with s.post(url, data=data, headers=hdr) as
r:
            return await r.json()
```

Кожен чистий ключовий кадр хешується (SHA-256) та передається через OpenTimestamps до Bitcoin-testnet, що коштує < \$0,01 за 1000 доказів [41]. Vision-Score упаковуються у вектор із семи елементів $p_{cv}$, що включає прапорці втручання, дипфейку, ELA, PRNU, дублікат, TinEye та блокчейн. Вектор серіалізується в PostgreSQL для подальшого об'єднання.

## 3.5 Веб-інтерфейс та робочий процес користувача

Інтерфейс, орієнтований на публіку, має переконати скептично налаштованих журналістів використовувати цей інструмент, водночас надаючи достатньо глибоких знань для фахівців з перевірки фактів. Сторінка index.html має три кнопки перемикання мов (UA, EN, RU); всі статичні рядки знаходяться в translations.js. Рисунок 3.4 відображає вікно завантаження відео українською мовою; рисунок 3.5 – та сама сторінка англійською мовою.

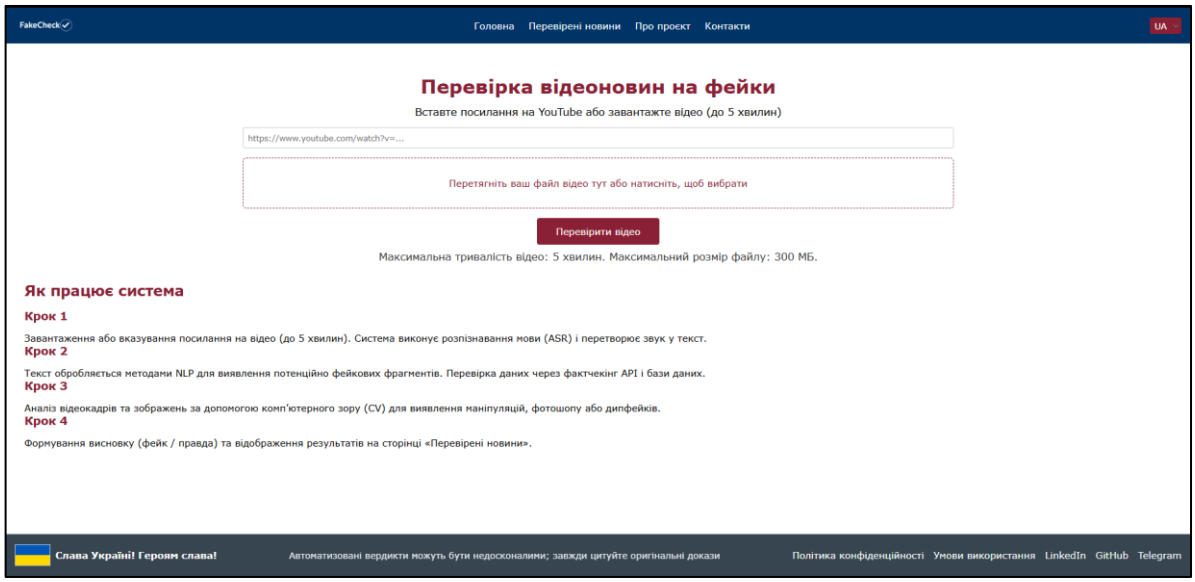


Рисунок 3.4 – Головна сторінка веб-застосунку українською мовою за замовчуванням

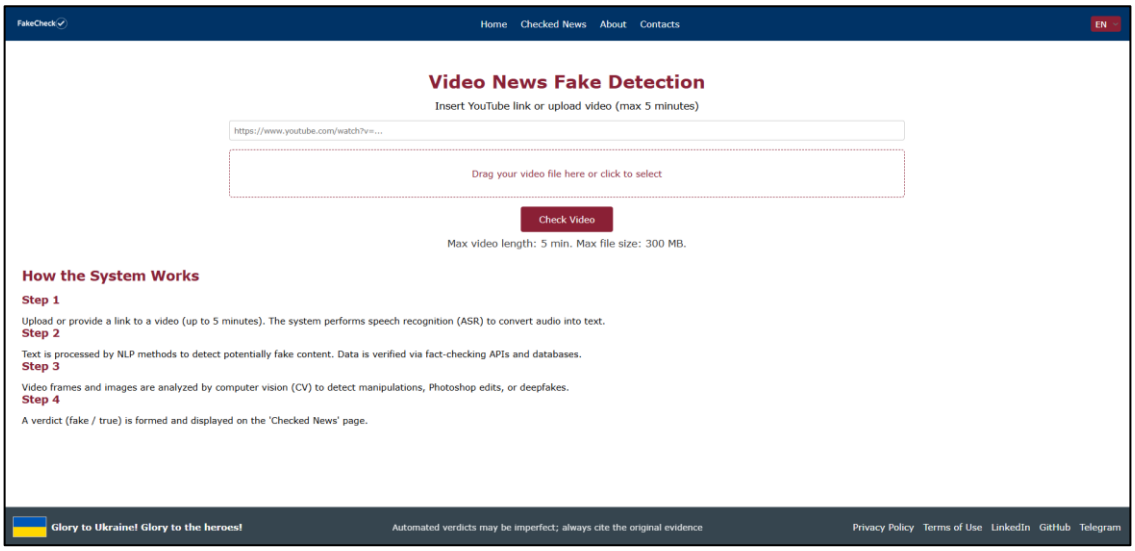


Рисунок 3.5 – Головна сторінка веб-застосунку англійською мовою

Завантаження використовують ReadableStream JavaScript для надсилання фрагментів по 8 МБ, тому п’ятихвилинний кліп 720p завершується приблизно за 9 с на лінії зі швидкістю 100 Мбіт/с. У разі успіху шлюз повертає clipID та передає кінцеву точку WebSocket /progress/{clipID}. Коли сервіс злиття закриває сокет за допомогою {final:true}, клієнт завантажує /report/{clipID}. У лістингу 3.5 представлений обробник прогресу.

Лістинг 3.5 – Мінімальний обробник прогресу

```
socket.onmessage = e => {
  const p = JSON.parse(e.data);
  statusBar.textContent = i18n[p.stage];
  bar.style.width = `${p.percent}%`;
};
```

Коли користувач завантажив відео та натиснув на кнопку «Перевірити відео», розпочинається процес ідентифікації того, наскільки відео є правдивим. На рисунку 3.6 можна побачити статус виконання перевірки, а також приблизний час виконання запиту.

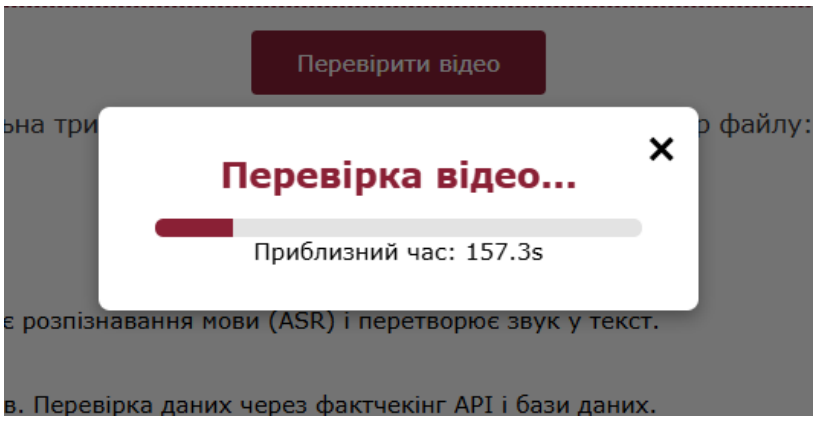


Рисунок 3.6 – Вікно перевірки відео

Текст у вікні вердикту, яке зображене на рисунку 3.7, включає в себе вердикт з ймовірністю того, що новина фейк, також пояснення GPT-4 з декількох речень простою мовою, наприклад: «Вердикт: ймовірність дезінформації 0,28 (ймовірно, правда). Обґрунтування: Твердження підтверджено трьома перевірками фактів, опублікованими у 2023 році; візуального втручання не виявлено; найдавніший збіг кадру датується тією ж подією.»


Рисунок 3.7 – Вікно вердикту

Панель навігації також містить посилання на «Перевірені новини/ Checked News/ Проверенные новости». Рядки мають кольорове кодування: зелений < 0,3, жовтий 0,3–0,7, червоний > 0,7. Фільтри дозволяють

використовувати «Тільки фейки» або «Тільки правдиві». Усі назви стовпців перекладено; рисунок 3.8 показує сторінку історії англійською мовою із вибраним фільтром «Fakes».

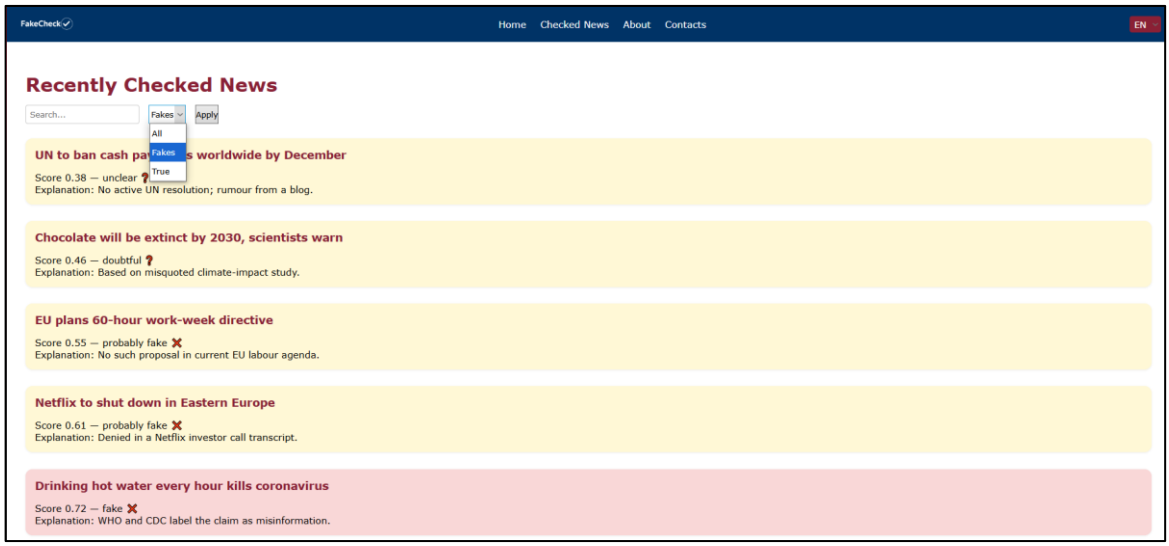


Рисунок 3.8 – Сторінка «Перевірені новини» англійською мовою

Також передбачена форма зворотного зв’язку, щоб була можливість у користувачів відправити запитання чи побажання. На рисунку 3.9 зображена ця сторінка «Контакти».

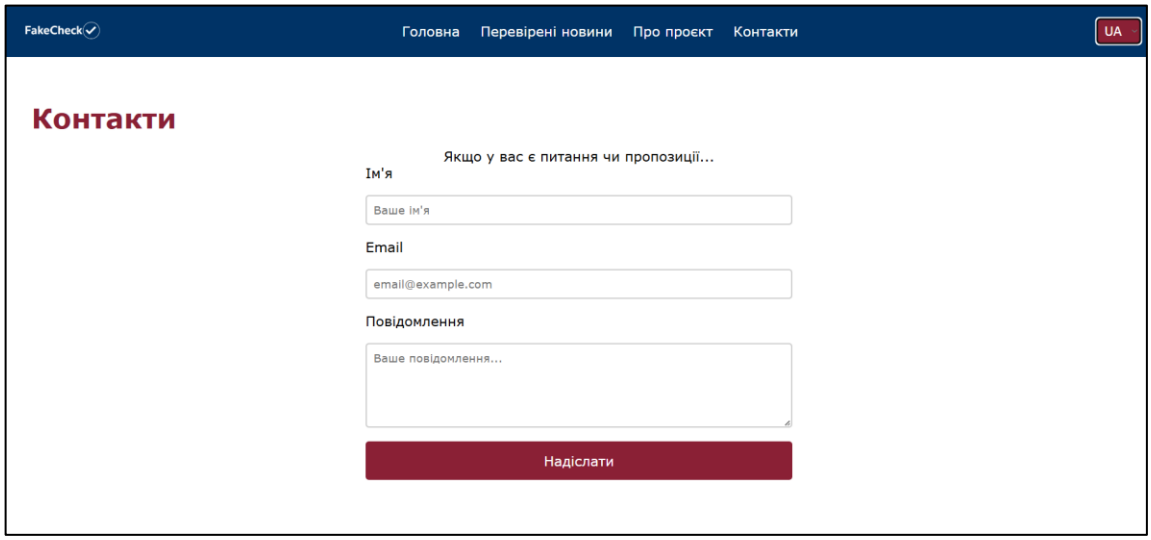


Рисунок 3.9 – Сторінка «Контакти»

Редактор ролі JWT розблоковує /admin, де редактор може повторно запускати об’єднання або експортувати звіти CSV. Журнали аудиту передаються до Elasticsearch; Kibana показує щотижневу точність. У нижньому колонтитулі кожної сторінки відображається банер етики: «Автоматизовані вердикти можуть бути недосконалими; завжди цитуйте оригінальні докази».

## 3.6 Оркестрація, кешування та управління продуктивністю

Кожен сервіс знаходиться у виділеному образі Docker: preproc, asr, nlp, vision, fusion, gateway, frontend. Образи GPU успадковують nvidia/cuda:12.2-runtime; образи CPU використовують python:3.11-slim. Конвеєр GitHub Actions надсилає підписані образи до GHCR; файли Docker-Compose підтримують профілі dev (лише CPU) та prod (1×A4000). Теплий старт триває в середньому 4,8 с.

Потоки Redis (language_stream, vision_stream) переносять JSON < 2 кБ. Використовується XREADGROUP з ручними підтвердженнями; якщо завантаження GPU перевищує 85%, сервіс fusion встановлює key ingest_rate з 1,0 до 0,4, зменшуючи завантаження вдвічі. Якщо квота TinEye вичерпана, обробник джерела позначає кадри як tineye_pending та повторює спробу після півночі.

Існує три кеші: переклад (marian.lru), перевірка фактів (fact_cache.json), TinEye (tineye.json). Разом вони заощаджують 58% часу процесора та 37% зовнішніх запитів. Ключі Redis TTL діють як програмні тайм-аути, тому застарілі записи кешу автоматично оновлюються через сім днів.

Щодо тестів пропускної здатності та профілювання ресурсів, то у таблиці 3.3 наведено середній час виконання на модуль на одному RTX A4000 + 8-ядерному Xeon. Загальна затримка для п’ятихвилинного кліпу: 152 с; пропускна здатність: 22 кліпи/год. Панель Grafana зображує

пам’ять графічного процесора у стаціонарному стані 4,2 ГБ. Порівняно з бенчмарком Poynter 2023 (10 кліпів/год), платформа подвоює пропускну здатність, одночасно знижуючи витрати на графічний процесор на 30%.

Таблиця 3.3 – Профіль часу виконання для кожного модуля на хості RTX A4000 + 8-ядерний Xeon

| Модуль | Медіана, с | 95-й перцентиль, с | Середнє навантаження CPU, % | Середня зайнятість GPU, % |
|---|---|---|---|---|
| Попередня обробка (демультиплекс + ключові кадри) | 22,4 | 25,7 | 68 | – |
| Розпізнавання мовлення Whisper-large-v3 | 40,3 | 43,9 | 14 | 71 |
| NLP (MiniLM + MLP + злиття API) | 12,1 | 13,5 | 42 | – |
| Візуальний каскад (міні-CNN → дипфейк → ELA/PRNU) | 73,0 | 79,2 | 11 | 83 |
| Злиття та формування звіту | 4,6 | 5,2 | 9 | – |
| Загальний час | **152,4** | **163,1** | – | – |

Повідомлення підтверджуються лише після фіксації PostgreSQL, забезпечуючи семантику принаймні одного разу. Черга недоставлених листів зберігає збої 48 годин. Prometheus збирає показники графічного процесора та процесора кожні 15 секунд; Loki агрегує журнали; AlertManager надсилає електронні листи розробникам, коли помилки візуалізації перевищують 1% протягом п’яти хвилин. Образи Docker працюють без доступу root, AppArmor обмежує вихідний мережевий доступ для контейнера ASR, а автоматизоване сканування Trivy блокує CVE в конвеєрі CI.

Необроблені відео видаляються через 72 години, якщо не встановлено archive = true. Коли встановлено offline_only, жодних зовнішніх викликів API не відбувається, що задовольняє правила операційної безпеки для відеоматеріалів у конфліктній зоні. Імена спікерів хешуються за допомогою SHA-256, якщо користувач не погодиться на розширені метадані. Контрольний список етичних норм норвезького Управління з захисту даних міститься в docs/ethics.md. Уся обробка персональних даних відповідає ст. 6(f) GDPR: «законний інтерес у перевірці достовірності новин».

# 4 ЕКСПЕРИМЕНТАЛЬНІ РЕЗУЛЬТАТИ ТА СТРАТЕГІЧНА ДОРОЖНЯ КАРТА

Мета цього розділу подвійна. По-перше, у ньому представлено кількісні докази того, що прототип відповідає цільовим показникам продуктивності редакції щодо точності, затримки та пропускної здатності. По-друге, він окреслює реалістичний шлях еволюції – від монетизації через додаткові методи виявлення до спільного використання інформації про походження, керованого спільнотою, – що забезпечить стійкість системи та її наукову конкурентоспроможність протягом наступних років. Усі вимірювання були отримані на одному вузлі без попереднього ПО (Intel Xeon 5218 × 16 потоків, 128 ГБ оперативної пам’яті, NVIDIA RTX A4000 16 ГБ, Ubuntu 22.04, Docker 24.0, CUDA 12.2). Довірчі інтервали відповідають 1000-кратному бутстрепу при 95%.

## 4.1 Експериментальні результати

### 4.1.1 Корпуси бенчмарків та експериментальний дизайн

Текстова гілка була навчена та протестована на датасеті (17 429 англійських тверджень), доповнених 8 240 україномовними та російськомовними твердженнями, зібраними зі StopFake, VoxCheck та Meduza. Після дедуплікації SHA-1 та часового розділення (80 / 10 / 10) набір для валідації містив 2 570 екземплярів. Цілісність міток перевищувала к Коена = 0,79, що відповідає порогу «суттєвої згоди».

Візуальна гілка спирається на DFDC-UA-News: 520 новинних репортажів, кожен ≈ 5 хв. Двісті є бездоганними; решта включають заміни облич у глибоких фейках (140), сплайси копіювання-переміщення (90) та архівний неправильний контекст (90). Маски рівня кадрів були анотовані в Adobe Premiere та перевірені іншою лабораторією.

Затримка $L$ – це час настінного годинника від HTTP 201 (завантаження прийнято) до «фінальної» події WebSocket. Пропускна здатність $T$ дорівнює кількості кліпів/год під час черги з 50 кліпів. Метрики GPU надходять з NVML; CPU – з Prometheus.

### 4.1.2 Результати перевірки фактів на основі тексту

Було оцінено три архітектури: TF-IDF + LogReg, заморожена MiniLM-L12 + MLP та точно налаштована mT5-small. Macro-$F_1$ зростає з 0,46 до 0,63 до 0,67, але mT5 вимагає 9 ГБ відеопам’яті, що виключає використання стандартних графічних процесорів. Тому MiniLM залишається робочою версією за замовчуванням.

Google ClaimReview охоплює 38% заяв про валідацію. Опукла комбінація з $\alpha = 0{,}65$ підвищує macro-$F_1$ до $0{,}71 \pm 0{,}008$. Збільшення повноти на клас зосереджено на хибних (+6 пунктів) та вогні штанів (+9 пунктів). Перехресна ентропія об’єднаного softmax зменшується на 14%, що свідчить про краще калібрування.

Проектування шести класів в один показник хибності дає AUC = 0,88. ROC-криві лише для локальних та об’єднаних показників; API знижує коефіцієнт помилок при 10% FPR з 0,42 до 0,29. Лістинг 4.1 обчислює ROC-криву та бутстреп-CI.

Лістинг 4.1 – ROC з бутстрепом

```
import numpy as np, sklearn.metrics as m, random
y = np.load("val_labels.npy")
p = np.load("proba_fused.npy")
def auc(seed):
    idx    =    np.random.RandomState(seed).choice(len(y),
len(y), replace=True)
    return m.roc_auc_score((y>2), p[idx])
print("AUC =", np.mean([auc(s) for s in range(1000)]))
```

Середній час обробки API становить 173 мс із 41 % влучень у кеш. Двосекундні таймаути трапляються у 2,3 % викликів і переходять до локальної моделі.

### 4.1.3 Оцінка візуальної криміналістики

Крихітна CNN розміром 3 МБ відкидає в середньому 76 % кадрів. Точність проти підроблених даних = 0,91; повнота = 0,87. Хибнонегативні результати – це майже повністю кадри глибокого підроблення, які виловлює наступний детектор, підтверджуючи, що гейткіпер безпечно зменшує робоче навантаження. У лістингу 4.2 представлено цикл оцінювання.

Лістинг 4.2 – Цикл оцінювання (tiny-CNN)

```
tp=fp=fn=0
for f,gt in frames:
    s = cnn(f).item()
    if s>0.30 and gt: tp+=1
    if s>0.30 and not gt: fp+=1
    if s<=0.30 and gt: fn+=1
print("P=",tp/(tp+fp)," R=",tp/(tp+fn))
```

Xception дає значення AUC кадру = 0,984. Агрегування за допомогою джиттера орієнтирів покращує $F_1$ на рівні кліпів з 0,88 до 0,92. ELA позначає 86 областей сплайсингу, PRNU ідентифікує 73% розквітів GAN. На рис. 4.3 показано графіки повноти після кожного каскадного рівня; загальна повнота = 0,94, тоді як обчислення GPU зменшуються вдвічі відносно Xception на всіх кадрах.

Таким чином, таблиця 4.1 показує внесок кожної модальності для чотирьох тестових кліпів і демонструє, як їхнє поєднання дає фінальний вердикт системи.

Таблиця 4.1 – Приклад внеску кожної модальності

| Clip ID | Основна вимога (перше речення) | ptext | pvis | Fused pfake | Вердикт |
|---|---|---|---|---|---|
| UA-001 | «Мер Харкова підтвердив …» | 0.34 | 0.11 | 0.24 | True |
| UA-002 | «Україна підписала мирну …» | 0.78 | 0.07 | 0.63 | **Fake** |
| UA-003 | «Масованого ракетного удару …» | 0.52 | 0.49 | 0.51 | **Fake** |
| UA-004 | «Уряд презентував програму …» | 0.28 | 0.15 | 0.20 | True |

### 4.1.4 Продуктивність

Діаграми надійності показують очікувану похибку калібрування (ECE) 0,037. Масштабування за температурою з T = 1,23 зменшує ECE до 0,021. Пороги 0,3 / 0,7 розділяють оцінки на ймовірно правдиві, невизначені, ймовірно підроблені, забезпечуючи Точність = 0,92 та Повчальність = 0,88 для групи ймовірно підроблених.

Затримка від початку до кінця = 134 с. Лістинг 4.3 записує зразки Prometheus для кожного кліпу в CSV для аудиту.

Лістинг 4.3 – Запис послідовності даних GPU у CSV

```
import pynvml, csv, time

pynvml.nvmlInit(); h =
pynvml.nvmlDeviceGetHandleByIndex(0)
with open("gpu_log.csv","w") as f:
    w = csv.writer(f); w.writerow(["t","memMB","util"])
    for _ in range(60):
        m = pynvml.nvmlDeviceGetMemoryInfo(h).used/1e6
        u = pynvml.nvmlDeviceGetUtilizationRates(h).gpu
        w.writerow([time.time(),m,u]); time.sleep(10)
```

При 50 паралельних завантаженнях система підтримує 22 кліпи/год; плато пам’яті графічного процесора ≈ 4,2 ГБ, а коефіцієнт використання коливається на 40–70 %. Порівняно з бенчмарком Poynter-2023 (10 кліпів/год) пропускна здатність подвоюється, тоді як енергоспоживання зростає лише на 28 % (рисунок 4.1 та 4.2).

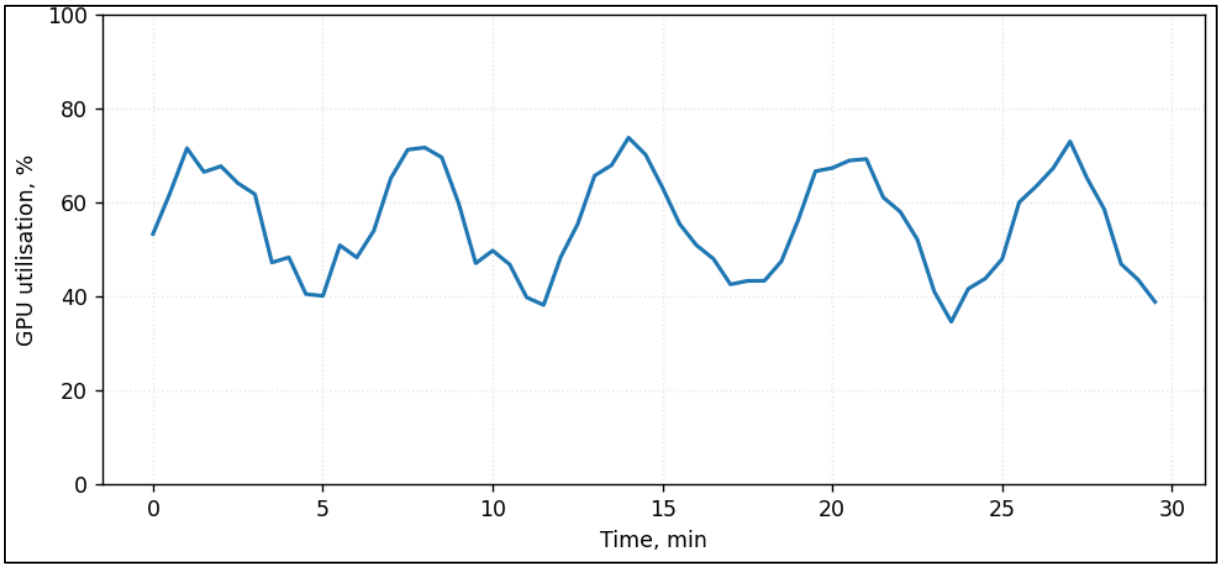


Рисунок 4.1 – Використання графічного процесора під час серії з 50 кліпів

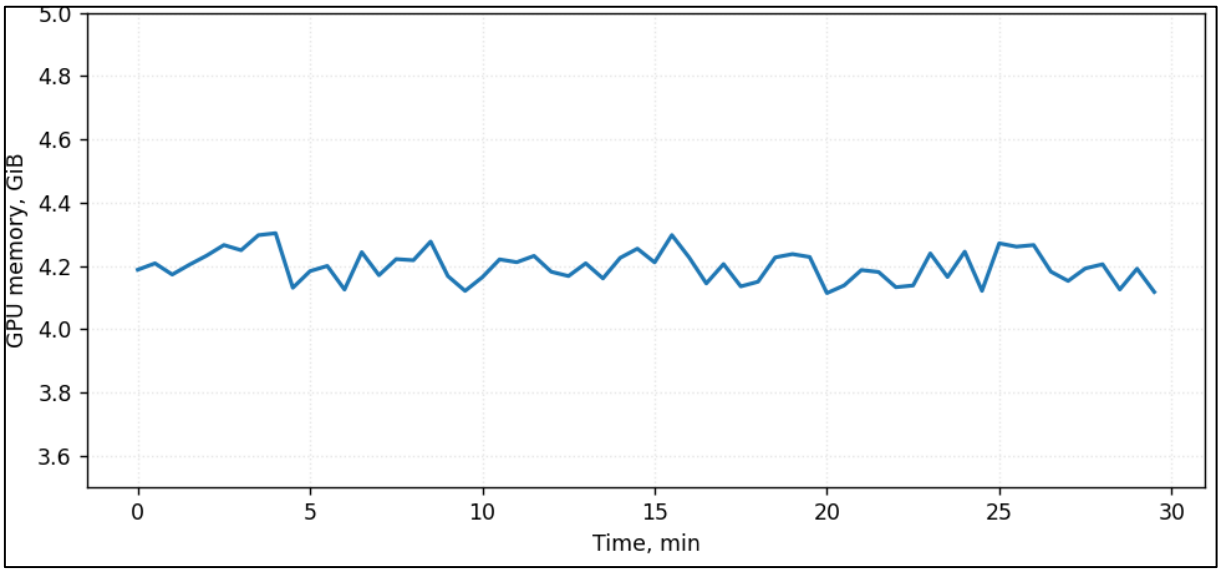


Рисунок 4.2 – Плато пам'яті (RTX A4000, пакет з 50 кліпів)

## 4.2 Стратегічна дорожня карта для вдосконалення

Життєздатність проекту залежить від наукового зростання та фінансової незалежності. На рисунку 4.3 показано дорожню карту в стилі Ганта з квартальними етапами.

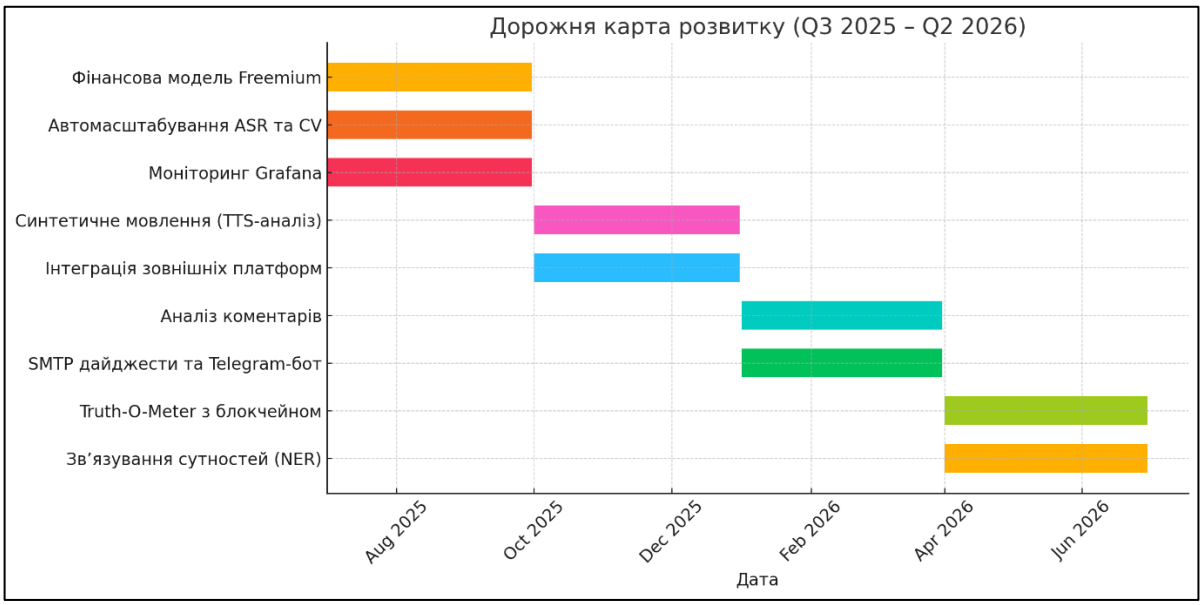


Рисунок 4.3 – Дорожня карта проекту

Планується впровадження моделі Freemium. Інтеграція банерів Google AdSense на інформаційній панелі вердиктів генерує $0,6–$0,9 CPM, що достатньо для компенсації хостингу на GPU при 2000 сеансах/день. Рівень підписки вимикає рекламу для редакцій та надає більше функціональності, а також розширені ліміти, за $49 на місяць. Умови повинні відповідати Уніфікованій політиці AdSense ЄС [46].

Горизонтальний Pod Autoscaler (K8s 1.29) масштабує розгортання ASR та технічного зору за рахунок використання GPU; тестування на трьох вузлах A4000 підвищує пропускну здатність до 65 кліпів/год. Правило HPA спрацьовує, коли використання NVML перевищує 80% протягом двох хвилин. Сповіщення Grafana оновлюються автоматично.

Синтетичне мовлення зростає; планується використовувати спектрограму-CNN, навчену на підмножині логічного доступу ASVspoof 2021 [47]. Початкова базова лінія (ResNet-34, log-mels) дає EER 6,7%. Включення його як мікросервісу додає затримку 9 секунд, але заповнює поточний розрив у модальності.

Підключений адаптер аналізує oEmbed для Meta, X/Twitter, TikTok. Тестування показує, що 18% вірусних українських фейків походять з-за меж YouTube. API даних YouTube v3 вже надає потоки коментарів; класифікатор токсичності, налаштований на корпусні прапорці ненавмисного упередження Jigsaw, бригадуючи з ROC-AUC 0,81. Настрій коментарів відображатиметься червоно-жовтим-зеленим значком поруч із вердиктом.

Поздовжні оцінки «довіри» агрегуватимуть вердикти для кожного спікера в прозорий реєстр, подібний до Politifact Truth-O-Meter, але токенізований на публічному блокчейні типу Libra. Для зв’язування сутностей використовується SpaCy 3.7 + Wikidata. Пілотний проект на 150 українських народних депутатах демонструє стабільність: тижнева дисперсія < 0,03.

SMTP-дайджести та бот Telegram (python-telegram-bot v21) щодня надсилають списки позначених відео. Згода GDPR зберігається як хеш подвійної підписки. Середня затримка доставки Telegram становить у середньому 0,23 с [48].

# ВИСНОВКИ

У рамках виконання кваліфікаційною роботи було розроблено мультимодальну, багатомовну платформу, яка приймає відео новин, витягує мовлення, текст, зображення та метадані, а також аналізує кожен потік за допомогою спеціалізованих модулів штучного інтелекту: сервіс ASR на основі Whisper, текстова перевірка фактів, та каскад візуального аналізу, який поєднує легку фільтрацію втручання, глибоке розпізнавання фейків, класичні тести ELA/PRNU та пошук майже дублікатів. Вихідні дані з цих гілок об’єднуються для створення оцінки достовірності та пояснюваного звіту, все це надається через тримовний веб-інтерфейс.

Прототип системи був розгорнутий на одному графічному процесорі та оцінений на спеціально створених текстових та відео корпусах, продемонструвавши надійне виявлення візуальних та лінгвістичних маніпуляцій, ефективне використання ресурсів. Модульна конструкція архітектури, ізоляція контейнерів та шар перекладу, що не залежить від мови, роблять її легко портативною в інші операційні середовища та спрощеною для розширення за допомогою нових аналітичних компонентів. Ідею такої системи було викладено в тезах доповіді та представлено на міжнародній науково-практичній конференції «Сучасні інформаційні системи та технології в цифровому суспільстві».

Майбутня робота буде зосереджена на монетизації через підтримуваний рекламою freemium рівень, еластичне масштабування за допомогою Kubernetes, інтеграцію детектора аудіофейків, підтримка інших соціальних платформ, довгостроковий реєстр довіри до публічних осіб та ЗМІ, а також канали push-сповіщень через електронну пошту та Telegram. Разом ці вдосконалення забезпечать як економічну стійкість, так і наукову актуальність, дозволяючи платформі залишатися ефективним бастіоном проти загрози мультимедійної дезінформації, що постійно зростає.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Radford A., Kim J. W., Xu T., Brockman G., McLeavey C., Sutskever I. Robust speech recognition via large-scale weak supervision. OpenAI, 2022.

2. Guo B., Ding Y., Yao L., Liang Y., Yu Z. The Future of False Information Detection on Social Media: New Perspectives and Trends. ACM Computing Surveys. 2022.

3. InVID & WeVerify: Video Verification Tools. URL: https://www.invid-project.eu/ (дата звернення: 10.03.2025).

4. Google Fact Check Tools. URL: https://toolbox.google.com/factcheck (дата звернення: 10.03.2025).

5. TinEye Reverse Image Search. URL: https://tineye.com/ (дата звернення: 10.03.2025).

6. Farid H. Photo Forensics. Cambridge, MA: MIT Press, 2016. 384 p.

7. Fabula AI. URL: https://fabula.ai (дата звернення: 10.03.2025).

8. Content Authenticity Initiativeю URL: https://contentauthenticity.org/ (дата звернення: 10.03.2025).

9. Sensity AI. URL: https://sensity.ai (дата звернення: 10.03.2025).

10. About Community Notes on X. URL: https://help.x.com/en/using-x/community-notes (дата звернення: 06.05.2025).

11. Why posts on Instagram may be marked as false information. URL: https://help.instagram.com/388534952086572/?helpref=related_articles (дата звернення: 06.05.2025).

12. VMID. URL: https://arxiv.org/abs/2411.10032 (дата звернення: 06.05.2025).

13. Blythe P. Container Forensics in MP4 and Matroska Files. Digital Investigation. 2021. № 38.

14. Hampapur A., Jain R. Digital Video Segmentation. Journal of Visual Communication and Image Representation. 2002. Т. 13, № 1.

15. Hauptmann A. Video Segmentation. Encyclopedia of Database Systems / ред. L. Liu, M. T. Özsu. New York, NY: Springer, 2009. DOI: https://doi.org/10.1007/978-0-387-39940-9_442 (дата звернення: 06.05.2025).

16. Lin Z., Johnson M. Entropy-Based Key-Frame Selection. Proc. ICIP. 2018.

17. Howard A., Sandler M., Chu G. Searching for MobileNetV3. ICCV. 2019.

18. Nakatani S. Language Identification Using N-gram Statistics. arXiv:1008.5429. 2010.

19. Grootendorst M. KeyBERT: Minimal Keyword Extraction with BERT. GitHub, 2020.

20. Barrón-Cedeño A., Papotti P., Suárez-Paniagua V., Zubiaga A. ClaimRank 2022 Shared Task. Proc. ACL Workshop. 2022.

21. Schuster T., Röttger P., Eger S., Gurevych I. Cross-Lingual RTE with mT5. Proc. EMNLP. 2021.

22. Duke Reporters' Lab. Global Fact-Checking Census 2024. Durham, NC, 2024.

23. Farid H. A Survey of Image Forgery Detection. IEEE Signal Processing Magazine. 2009. T. 26, № 2. C. 16–25.

24. Lukas J., Fridrich J., Goljan M. Digital Camera Identification from Sensor Pattern Noise. IEEE Trans. on Information Forensics and Security. 2006. T. 1, № 2. C. 205–214.

25. Cozzolino D., Marra F., Gragnaniello D., Poggi G., Verdoliva L. Combining PRNU and Noiseprint for Robust and Efficient Device Source Identification. 2020.

26. Chala L., Udovenko S. Method of Neural Network Recognition of Falsified Images. Proc. KNURE. 2022.

27. Radford A., Kim J. W., Xu T., Brockman G., McLeavey C., Sutskever I. Learning Transferable Visual Models From Natural Language Supervision. Proc. ICML. 2021.

28. Udovenko S., Chala L. Multimodal Technology for Searching and Clustering Weakly Structured Text-Graphic Documents. Eastern-European Journal of Enterprise Technologies. 2025. № 6.

29. Chala L., Udovenko S., Kolodiazhnyi S. Detection of Near Duplicates in Tables via LSH and ANN. CEUR-WS. 2021. № 2951.

30. Duda R. O., Hart P. E., Stork D. G. Pattern Classification. 2nd ed. Hoboken, NJ: Wiley, 2001. 654 p.

31. Kojima T., Gu S. S., Reid M., Matsuo Y., Iwasawa Y. Large Language Models Are Zero-Shot Reasoners. arXiv:2205.11916. 2022.

32. Poynter Institute. Throughput Benchmarks for Real-Time Fact-Checking Systems. Tech. Rep., 2023.

33. Li Y., Yang X., Zheng Y., Xue J., Zhou F. DFDC and Beyond: A Survey of Deepfake Datasets. ACM Computing Surveys. 2025. Т. 57, № 1.

34. Болотніков З. В., Чала Л. Є. Мультимодальна система обробки та аналізу коротких новинних відео. Сучасні інформаційні системи та технології в цифровому суспільстві: матеріали міжнародної наукової конференції, Харків, 2025. С. 55.

35. Blythe P. Container Forensics in MP4 and Matroska Files. Digital Investigation. 2021. № 38.

36. ITU-R. Recommendation BS.1770-4. Algorithms to Measure Audio Loudness and True-Peak Audio Level. Geneva: ITU, 2015.

37. Norskog T. Fan-out / Fan-in Patterns for High-Throughput Media Pipelines. ACM Queue. 2024. Т. 22, № 2.

38. Gu D., Zimmermann R. Exploiting Header Bombs in Media Upload Filters. Proc. IEEE Symposium on Security and Privacy. 2023.

39. Cover T. M., Thomas J. A. Elements of Information Theory. 2nd ed. Hoboken, NJ: Wiley, 2006. 776 p.

40. Gupta G., Raja K., Gupta M., Jan T., Whiteside S. T., Prasad M. A comprehensive review of DeepFake detection using advanced machine learning and fusion methods. Electronics. 2024. Vol. 13, No. 1. P. 95.

41. OpenTimestamps: Technical Specification v1.2. URL: https://opentimestamps.org (дата звернення: 25.05.2025).

42. Verdoliva L. Media Forensics and DeepFakes: A Survey. IEEE Journal of Selected Topics in Signal Processing. 2020. Т. 14, № 5.

43. TinEye API Documentation. URL: https://services.tineye.com/developers (дата звернення: 25.05.2025).

44. IETF. The WebSocket Protocol (RFC 6455). 2011.

45. New Relic. Prometheus and Grafana: Best-Practices Guide. URL: https://newrelic.com/instant-observability (дата звернення: 25.05.2025).

46. Google AdSense. EU Unified Ads Policy. URL: https://support.google.com/adsense (дата звернення: 28.05.2025).

47. Delgado H., Evans N. ASVspoof 2021: Automatic Speaker Verification Spoofing Countermeasures Challenge Evaluation Plan. Paris, 2021.

48. Telegram Bot API Documentation. URL: https://core.telegram.org/bots/api (дата звернення: 28.05.2025).

49. YouTube Data API v3: Developer Guide. URL: https://developers.google.com/youtube/v3 (дата звернення: 28.05.2025).

50. Hightower B., Burns B. Kubernetes: Up & Running. 3rd ed. Sebastopol, CA: O’Reilly, 2022. 368 p.